UNITED STATES COURT OF APPEALS FOR THE FEDERAL CIRCUIT

VIRNETX INC.,
Appellant

v.

APPLE INC.,
Appellee

2022-1523

Appeal from the United States Patent and Trademark Office,
Patent Trial and Appeal Board in No. 95/001,682

OPENING BRIEF FOR APPELLANT VIRNETX INC.

Naveen Modi
Joseph E. Palys
Stephen B. Kinnaird
Igor V. Timofeyev
Daniel Zeilberger
PAUL HASTINGS LLP
2050 M Street, N.W.
Washington, D.C. 20036
(202) 551-1700
naveenmodi@paulhastings.com

Counsel for Appellant VirnetX Inc.

August 22, 2022

SELECTED CLAIM AT ISSUE

U.S. Patent No. 6,502,135

Claim 18:

A method of transparently creating a virtual private network (VPN) between a client computer and a target computer, comprising the steps of:

(1) generating from the client computer a Domain Name Service (DNS) request that requests an IP address corresponding to a domain name associated with the target computer;

(2) determining whether the DNS request transmitted in step (1) is requesting access to a secure web site; and

(3) in response to determining that the DNS request in step (2) is requesting access to a secure target web site, automatically initiating the VPN between the client computer and the target computer, wherein:

steps (2) and (3) are performed at a DNS server separate from the client computer, and step (3) comprises the step of, prior to automatically initiating the VPN between the client computer and the target computer, determining whether the client computer is authorized to resolve addresses of non secure target computers and, if not so authorized, returning an error from the DNS request.

CERTIFICATE OF INTEREST

Counsel for Appellant VirnetX Inc. certifies the following:

1. The full name of the party represented by me in this case is:

   VirnetX Inc.

2. The name of the real party in interest represented by me is:

   None

3. All parent corporations and any publicly held companies that own 10 percent or more of the stock of the party represented by me are:

   VirnetX Holding Corporation

4. All law firms, partners, and associates that appeared for the party represented by me in the agency below, or are expected to appear in this Court (and who have not or will not enter an appearance in this case):

   None

5. The title and number of any case known to counsel to be pending in this Court or any other court or agency that will directly affect or be directly affected by this Court's decision in the pending appeal:

   VirnetX Inc. v. Cisco Systems, Inc., No. 19-1671 (Fed. Cir.);

   VirnetX Inc. v. Mangrove Partners Master Fund,
   Nos. 20-2271, -2272 (Fed. Cir.);

   VirnetX Inc., Leidos, Inc. v. Apple Inc., No. 21-1672 (Fed. Cir.);

   VirnetX Inc. v. Apple Inc., No. 6:13-cv-00211(E.D. Tex.).

6. Information required by Federal Rule of Appellate Procedure 26.1(b) and (c) that identifies organizational victims in criminal cases and debtors and trustees in bankruptcy cases:

   Not Applicable

Dated: August 22, 2022

/s/Naveen Modi
Naveen Modi
Counsel for Appellant VirnetX Inc.

TABLE OF CONTENTS

Page

TABLE OF AUTHORITIES

Page(s)

Cases

Statutes

Rules and Regulations

Other Authorities

STATEMENT OF RELATED CASES

This is an appeal from an *inter partes* reexamination proceeding before the Patent Trial and Appeal Board ("the Board"). In the proceeding below, the Board issued a decision finding claims 15 and 16 of U.S. Patent No. 6,502,135 ("the '135 patent") patentable but claims 10-14, 17, and 18 of the '135 patent unpatentable.

There are several other (prior or pending) appeals involving the patent at issue (the '135 patent). In No. 17-1368, the Court considered an appeal from the Board's decision in an *inter partes* review proceeding IPR2015-01046 that found several claims of the '135 patent (and those of a related patent) to be invalid over a different prior art reference (Kiuchi) than the references at issue in this appeal. This Court held that the Board erred in mapping the claim term "client computer" in the '135 patent to Kiuchi's client-side proxy without considering VirnetX's proposed construction of "client computer." *VirnetX Inc. v. The Mangrove Partners Master Fund, Ltd.*, 778 F. App'x 897, 908-09 (Fed. Cir. 2019) (Moore, J., joined by Prost, C.J., and Reyna, J.). The Court also held that the Board improperly refused to give effect to VirnetX's prosecution disclaimer when construing the term "virtual private network" in the '135 patent. *Id.* at 910. The Court vacated the Board's unpatentability findings and remanded to the Board with instruction "to assess Kiuchi's disclosure in light of the proper construction." *Id.*

On remand, the Board agreed with VirnetX's proposed construction of "client computer" and accepted that, under this Court's ruling in *Mangrove*, the term "virtual private network" must be interpreted to exclude non-direct communications. The Board, however, nevertheless found the claims at issue to be unpatentable. VirnetX's appeal from that decision is pending before this Court as No. 20-2271.

The '135 patent was also at issue in an *inter partes* reexamination proceeding No. 95/001,679. In that proceeding, the Board found claims 1-9 and 13-18 of the '135 patent unpatentable. VirnetX appealed that decision to this Court, and that appeal is currently pending as No. 19-1671. In the No. 19-1671 appeal, in the interest of judicial efficiency, VirnetX did not challenge the Board's determination regarding claims 13-17.

This Court also considered (or is considering) several appeals from district court proceedings that involved the '135 patent (and related patents). The first appeal (No. 13-1489) was from an infringement proceeding that VirnetX initiated in the United States District Court for the Eastern District of Texas in 2010. After a jury trial, the district court entered a judgment upholding the asserted claims against an invalidity challenge. The jury also found that Appellee Apple Inc. ("Apple")—one of the defendants in the action—infringed the patents-in-suit, and awarded damages. *VirnetX Inc. v. Apple Inc.*, 925 F. Supp. 2d 816 (E.D. Tex. 2013). This Court affirmed-in-part, reversed-in-part, vacated-in-part, and remanded for further

proceedings. *VirnetX, Inc. v. Cisco Sys., Inc.*, 767 F.3d 1308 (Fed. Cir. 2014) (Prost, C.J., joined by Chen, J.).

On remand, after another jury trial, the district court entered a judgment again finding infringement by Apple and awarding damages. In a subsequent appeal (No. 18-1197), this Court affirmed that judgment under Federal Circuit Rule 36. *VirnetX Inc. v. Cisco Sys., Inc.*, 748 F. App'x 332 (Fed. Cir. 2019) (per curiam) (Prost, C.J., Moore and Reyna, JJ.). The Supreme Court denied a petition for a writ of certiorari. *Apple Inc. v. VirnetX Inc.*, 140 S. Ct. 1122 (2020).

This Court also considered an appeal from a different infringement proceeding VirnetX initiated against Apple in the United States District Court for the Eastern District of Texas in 2012, asserting the '135 patent (and three other patents). In that proceeding, the district court granted VirnetX a summary judgment on invalidity, finding that Apple was precluded from raising its invalidity challenges because of prior litigation. After a jury trial, the district court entered a judgment for VirnetX finding infringement by Apple and awarding damages. *VirnetX Inc. v. Apple Inc.*, No. 12-cv-00855, 2018 WL 10048706 (E.D. Tex. Aug. 30, 2018). On appeal (in No. 19-1050), this Court upheld the district court's ruling that Apple was precluded from making its invalidity challenges and upheld the finding of infringement as to the '135 patent (and one related patent). The Court, however, reversed the judgment of infringement as to the other two patents at issue, and

therefore vacated the damages award and remanded. *VirnetX Inc. v. Apple Inc.*, 792 F. App'x 796 (Fed. Cir. 2019) (Taranto, J., joined by Mayer and Lourie, JJ.).

On remand, the district court held a new trial on damages. The jury awarded VirnetX over $502 million in damages for Apple's infringement of the '135 patent (and a related patent), and the district court entered a final judgment. Apple's appeal from that judgment is currently pending before this Court as No. 21-1672.

INTRODUCTION

Appellant VirnetX Inc. ("VirnetX") pioneered easy-to-use methods of secure communications over the Internet. In contrast to the prior art, which required special software and a complex set-up process, VirnetX's patented methods and devices leverage domain names in a new way that enables users to establish secure communications with ease. Appellee Apple Inc. ("Apple") uses this technology in well-known applications like FaceTime, and this Court has previously affirmed multiple jury findings that Apple infringes the patent at issue—U.S. Patent No. 6,502,135 ("the '135 patent").

Claim 18 of the '135 patent is generally directed to the formation of "a virtual private network (VPN)" between a client computer and a target computer. Among other things, it recites a particular way of checking and acting upon client authorization. Specifically, claim 18 requires, "prior to automatically initiating the VPN between the client computer and the target computer, determining whether the client computer is authorized to resolve addresses of non secure target computers and, if not so authorized, returning an error from the DNS request." Appx134 (2:16-21). In the proceeding below, the Patent Trial and Appeal Board ("the Board") did not even pretend to interpret the words of the claim or analyze whether the prior art met the claim's limitations. Instead, the Board proclaimed that "claim 18 is confusing because the language appears to be directed to the disclosed scenario [in

the specification], but the claim language does not clearly recite this scenario." Appx31. Based on that professed confusion about the relationship of the claim to the specification, the Board announced that it would only "decide whether as a general matter, the prior art would have rendered obvious determining whether a client has permission to access a web site and if not, returning an error message." Appx31.

The Board's approach, which erased various requirements from the authorization limitation of claim 18, contravenes this Court's consistent instruction that "[t]he claims, not specification embodiments, define the scope of patent protection." *Kara Tech. Inc. v. Stamps.com Inc.*, 582 F.3d 1341, 1347-48 (Fed. Cir. 2009). It was error for the Board to ignore limitations in claim 18 that it could not find in the specification. Moreover, the Board's premise was itself faulty—the authorization limitation at issue in claim 18 *is* found in the specification, specifically in the original claims filed with the application that led to the '135 patent. "Original claims are part of the specification and in many cases will satisfy the written description requirement." *Crown Packaging Tech., Inc. v. Ball Metal Beverage Container Corp.*, 635 F.3d 1373, 1380 (Fed. Cir. 2011). An examination of claim 18's actual language, rather than the Board's simplified version of it, makes it readily apparent that the prior art does not meet the claim.

The Board committed other errors as well. In addressing other limitations of claim 18, the Board applied collateral estoppel to VirnetX's arguments with respect to rejections based on the Beser reference, even though the prior proceedings the Board identified involved different claim language and different issues.[1] And in finding that the BinGO reference anticipated various claims (including claim 18), the Board improperly relied on the combination of two separate pieces of prior art, erroneously thinking that a mere reference in one prior art document to another prior art document constitutes an incorporation by reference. But "a mere *reference* to another application, or patent, or publication is not an *incorporation* of anything therein." *In re de Seversky*, 474 F.2d 671, 674 (C.C.P.A. 1973). Rather, in order "[t]o incorporate material by reference, the host document must identify with detailed particularity what specific material it incorporates and clearly indicate where that material is found in the various documents." *Advanced Display Sys., Inc. v. Kent State Univ.*, 212 F.3d 1272, 1282 (Fed. Cir. 2000).

The Board's final written decision should be reversed or, at a minimum, vacated and remanded.

[1] The Board did not apply collateral estoppel with respect to the authorization limitation.

JURISDICTION

The Board issued its final decision in the *inter partes* reexamination below on May 14, 2021, Appx1-53, and denied rehearing on January 10, 2022, Appx54-57. VirnetX timely appealed to this Court on March 10, 2022. Appx7064-7067. This Court has jurisdiction under 35 U.S.C. § 141(b) and 28 U.S.C. § 1295(a)(4)(A).

ISSUES PRESENTED

1. Whether the Board erred in analyzing the authorization limitation in claim 18 of the '135 patent when it:
   (a) ignored claim limitations that it could not find in the specification, and
   (b) incorrectly found, based on the faulty construction, that the asserted references taught the limitation.
2. Whether the Board erroneously applied collateral estoppel to VirnetX's other arguments with respect to the rejections based on the Beser reference.
3. Whether the Board erroneously relied on a combination of references in finding anticipation based on the BinGO reference.
4. Whether the Board failed to properly consider VirnetX's petition for rehearing.

STATEMENT OF THE CASE

I. THE TECHNOLOGY AT ISSUE

The invention at issue relates to secure Internet communications. Communications are sent using an Internet Protocol (IP) address, a four-segment

string of binary numbers (often represented in decimal form, e.g., "19.28.37.156"). Appx118 (37:22-29). When one device sends data to another over the Internet, the data is broken up into "packets." The packets are labeled with the IP addresses associated with the sending and destination computers. The packets travel across the Internet using a series of specialized devices called "routers," which direct traffic over the networks comprising the Internet. Appx65 (Fig. 1). Each router checks the packet's destination IP address against internal tables, and passes the packet to the next router. Packets traverse the Internet's hierarchy of networks and routers until they are ultimately delivered to the destination computer.

Internet communications are ordinarily nonsecure. Packets can be intercepted during routing by hackers, who can then read the data within. Appx118 (37:40-49). The meteoric upswing in Internet use in the late 1990s generated demand for easy and secure network communications. Appx100 (1:10-13, 1:16-18).

Before VirnetX's inventions, reliable secure communications were primarily achieved through encrypted Virtual Private Networks ("VPNs"). Appx4326-4327. VPNs, however, were cumbersome to use. VPN users had to be trained to set up encryption keys used to scramble and unscramble the data, and the sender and the receiver had to configure multiple parameters for the VPN link. VPN remote access was "a nightmare for [IP] support desks." Appx4326. The complexity of VPNs presented an unintended security risk from incorrect use. Appx4326-4327.

The '135 patent provided an innovative alternative that generated VPNs automatically. The inventors looked to the domain name system, which translates user-friendly "domain names," such as "yahoo.com," into IP addresses that routers can use to direct data to a device, such as "98.137.11.163." For example, when a user types a domain name into a web browser, the user's device sends a request to a Domain Name Server ("DNS"). The DNS responds with the IP address corresponding to that domain name, which allows the user's device to connect with the target computer. Appx118 (37:22-29). VirnetX's inventions allow users to employ familiar techniques—typing a domain name into the browser—whereupon the system seamlessly establishes a secure VPN between two devices. These inventions provided an easy method of ensuring data security, particularly for business travelers or between private networks. *See, e.g.*, Appx100 (2:62-64); Appx102 (6:1-3).

II. THE '135 PATENT

The '135 patent relates to a system and method in which a DNS proxy automatically and transparently creates a VPN in response to a DNS lookup. *See VirnetX, Inc. v. Cisco Sys., Inc.*, 767 F.3d 1308, 1315 (Fed. Cir. 2014); Appx118-119 (37:17-39:1); Appx93-94 (Figs. 26-27).

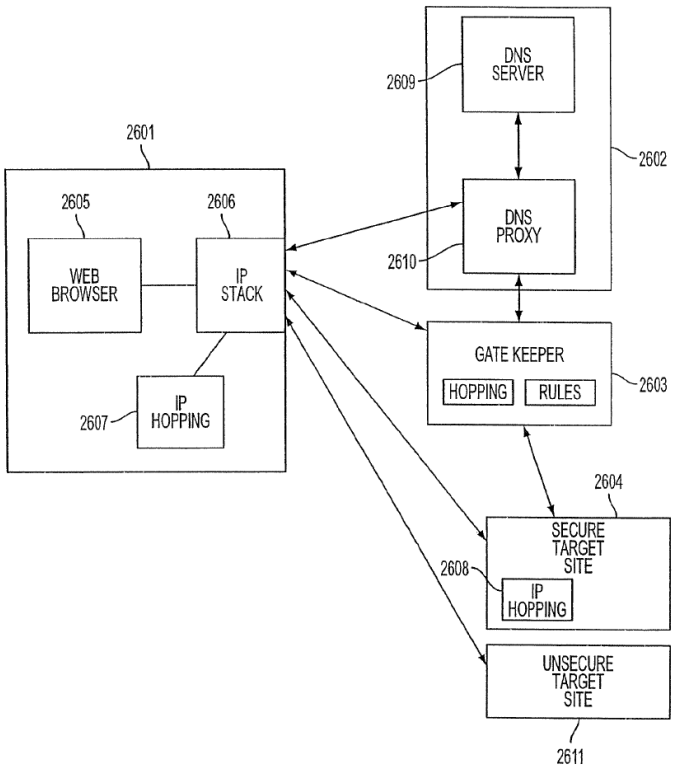


FIG. 26

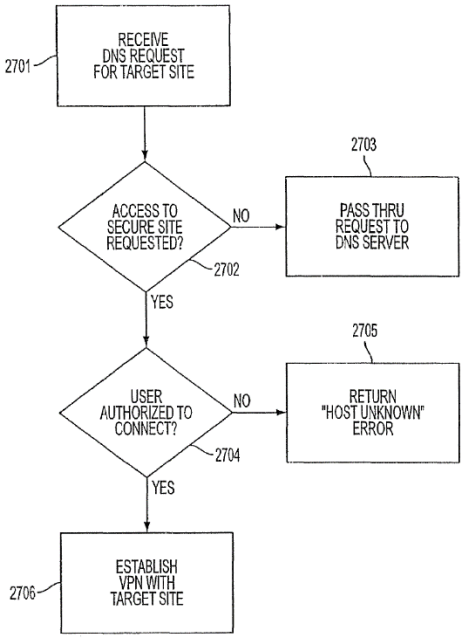


FIG. 27

As shown in Figures 26 and 27, browser (2605) within user computer (2601) generates a DNS request for an IP address corresponding to a domain name of a target computer, such as secure target site 2604 and/or unsecure target site 2611. Appx118-119 (38:23-39:6). Instead of a conventional DNS server (2609) receiving the request, DNS proxy (2610) intercepts the request and determines whether it is for a secure website. Appx118-119 (38:23-25, 39:2-3). According to a preferred embodiment, the determination is made by checking the domain name against domain name tables. Appx118 (38:23-30).

If the domain name is listed therein, DNS proxy (2610) determines that the user computer is requesting access to a secure site, and may determine whether the

user and/or computer 2601 is authorized to access the secure web site. Appx118-119 (38:23-30, 39:7-20). If so, DNS proxy (2610) automatically initiates a VPN between browser (2605) and secure target site (2604). Appx118 (38:25-33).

If the domain name is *not* listed in the domain name table, DNS proxy (2610) determines that the request is *not* seeking access to a secure site and forwards the request to conventional DNS (2609) without initiating a VPN. Appx118-119 (38:43-47, 39:3-6). This contrasts with a situation where, for example, an unauthorized user has requested lookup of a secure site, which would instead cause the DNS proxy to return a "host unknown" error to the user. Appx118 (38:47-50).

One embodiment for encrypting VPNs is the Tunneled Agile Routing Protocol ("TARP"). A client directly addresses a target using the IP address of the target. Appx103 (7:40-8:36). To avoid detection during routing, however, this target IP address "is concealed behind an outer layer of encryption generated using a link key." Appx103 (7:59-60). TARP routers sit between the client and target and "can use [a] link key to reveal the true destination of a TARP packet." Appx103 (7:40-8:1). Based on a "time to live counter" designed to "help foil traffic analysis," a TARP router "determine[s] … whether it should forward the TARP packet [] to another TARP router [] or to the destination TARP terminal." Appx103 (8:16-36).

During prosecution, VirnetX disclaimed any VPN that does not require direct communication. As this Court previously observed, VirnetX distinguished "a

system in which a client computer communicates with an intermediate server via a singular, point-to-point connection," whereby "[t]hat intermediate server then relays the data to a target computer on the same private network on which the server resides." *VirnetX Inc. v. The Mangrove Partners Master Fund, Ltd.*, 778 F. App'x 897, 910 (Fed. Cir. 2019). VirnetX explained that, because in such a system "the computers 'do not communicate directly with each other' and '[t]he client cannot open a connection with the target itself,' the computers are not on the same VPN." *Id.* (citation omitted). This disclaimer, the Court concluded, "clearly and unmistakably states that a 'VPN between the client computer and the target computer' requires direct communication between the client and target computers." *Id.*

Claim 18 of the '135 patent is representative, and recites:

> A method of transparently creating a virtual private network (VPN) between a client computer and a target computer, comprising the steps of:
>
> (1) generating from the client computer a Domain Name Service (DNS) request that requests an IP address corresponding to a domain name associated with the target computer;
>
> (2) determining whether the DNS request transmitted in step (1) is requesting access to a secure web site; and
>
> (3) in response to determining that the DNS request in step (2) is requesting access to a secure target web site, automatically initiating the VPN between the client computer and the target computer, wherein:

> steps (2) and (3) are performed at a DNS server separate from the client computer, and step (3) comprises the step of, prior to automatically initiating the VPN between the client computer and the target computer, determining whether the client computer is authorized to resolve addresses of non secure target computers and, if not so authorized, returning an error from the DNS request.

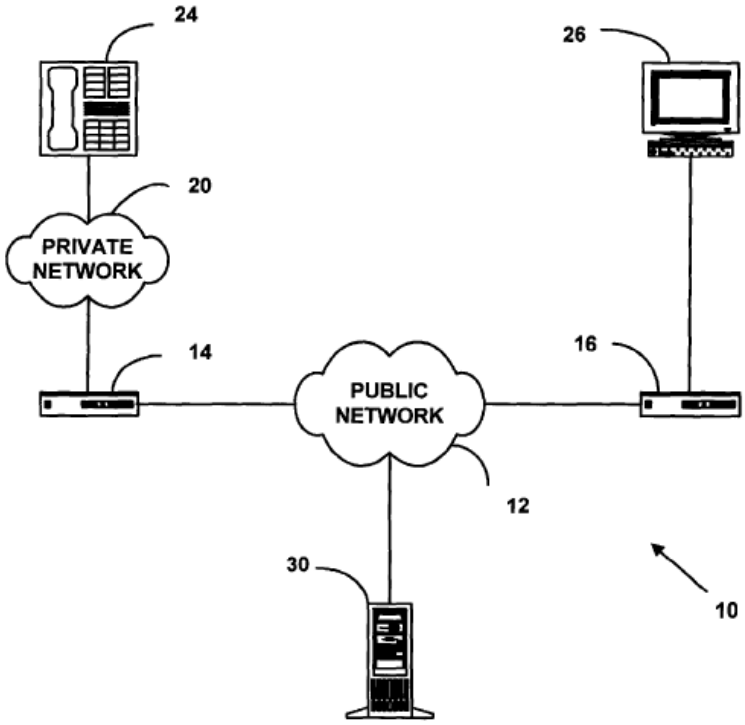
Appx134 (claim 18).

III. THE ASSERTED REFERENCES

A. Beser

U.S. Patent No. 6,496,867 to Beser *et al.* ("Beser") "relates to communications in data networks," Appx2255 (1:8-9), and recognizes that "the Internet is not a very secure network," Appx2255 (1:26-27). Beser teaches a "tunneling association" that hides the originating and terminating ends of the tunnel during communications on a public network. Appx2256 (3:1-9). Because the source is hidden, hackers are prevented from intercepting communications, resulting in "increase[d] security of communication without an increased computational burden." Appx2255-2256 (2:36-40, 3:4-9).

Beser's solution involves "initiating a tunneling association between an originating end [24] and a terminating end [26]" facilitated by an intermediary, trusted-third-party network device 30. Appx2255-2258 (1:45-67, 7:62-64). Figure 1 of Beser illustrates this solution:



Appx2238 (Fig. 1).

When an originating end device 24 in Beser wants to communicate with a terminating end device 26, it sends a tunnel initiation request 112 to first network device 14. Appx2258 (7:65-67); Appx4099 (¶¶ 88, 89). This request "includes a unique identifier for the terminating end of the tunneling association." Appx2258 (8:1-3); *see also* Appx4099 (¶¶ 88, 89).

The first network device 14 then sends an inform message 114 with tunnel initiation request 112 to trusted-third-party network device 30 by constructing one or more IP packets 58. Appx2258 (8:3-4); Appx2260 (11:9-25). The trusted-third-party network device 30 associates a public IP address of a second network device 16 with the unique identifier of terminating telephony device 26. Appx2258 (8:4-

7); Appx2260 (11:26-32); Appx4099 (¶ 89). The first and second network devices 14 and 16 then "negotiate" private IP addresses through the public network 12. Appx2243 (Fig. 6 (step 118)); Appx2258 (8:9-15); Appx2260 (11:58); *see also* Appx4099 (¶ 89). This "negotiation" assigns a first private network address to the originating device 24 and a second private network address to the terminating device 26. Appx2260 (12:2-4).

Once assigned, the private network address of originating device 24 and the public IP address of first network device 14 are communicated to the second network device 14. Appx2261 (13:33-48); Appx4099 (¶ 89). Similarly, the private network address of the terminating device 26 and the public IP address of the second network device 16 are communicated to the first network device 14. Appx2261 (14:19-33); Appx4099 (¶ 89).

B. Kent

Kent is a Request for Comments document, RFC 2401, that describes IPSec, a type of security protocol for the Internet. Appx2272; Appx4099-4100 (¶ 91). RFC 2401 describes a method for automatically encrypting network traffic that is sent through security gateways over a public network, and provides end-to-end encryption. *See, e.g.*, Appx2296-2299. Kent also describes the existence of "ICMP error messages," which are generally related to providing a message when a service

is not available or a host or router could not be reached. *See, e.g.*, Appx2308; Appx4105 (¶ 110).

C. Blum

U.S. Patent No. 6,182,141 to Blum *et al.* ("Blum") is directed to a method for providing transparent proxy services. Appx2348 (2:25-26); Appx4104 (¶ 108). A "proxy server," as defined by Blum, is an application that provides access to the Internet or other external network, and evaluates requests and determines which of the requests to pass on to the Internet or other network. Appx2348 (1:10-25); Appx4104 (¶ 108). Blum enables communications from a client computer, through a proxy, to remote DNS. Appx2350 (6:40-43); Appx4104 (¶ 108). Blum describes returning an error message if a DNS request fails because DNS services available to the client computer were not able to resolve the request. Appx2351 (8:65-9:3); Appx4105 (¶ 110).

D. BinGO

BinGO generally refers to two separate documents that Apple presented as a "single reference document": a printed publication titled "BinGO! User's Guide" ("BinGO UG"), purportedly published by March 1999, and another printed publication titled "Extended Feature Reference" ("BinGO EFR"), purportedly published by February 1999. Appx147.

BinGO UG is directed to a router used to connect a user with the network of an Internet provider or to a company's head office from the user's home or branch office via an integrated services digital network (ISDN). Appx2369-2370; Appx4105-4106 (¶ 113). BinGO UG states that if the user wants to access the Internet, the user must set up the user's Internet service provider (ISP) as a wide area network (WAN) partner on the BinGO router, and if the user wishes to establish a local area network (LAN) to LAN connection (e.g., between the user's LAN and the LAN of the user's head office), the user must configure the LAN of the user's head office as a WAN partner. Appx2508; Appx4105-4106 (¶ 113). BinGO further describes how to configure the BinGO router (e.g., by using a configuration wizard) so that it may be connected to a WAN partner such as a corporate network. Appx2399; Appx2418-2419; Appx4105-4106 (¶ 113).

In configuring the BinGO router, encryption for a connection to the WAN partner may be selected. Appx2514-2515; Appx2540; Appx4106 (¶ 114). While BinGO UG describes that the connection to the WAN partner (e.g., either the ISP or the corporate network) may be encrypted, the additional reference BinGO EFR describes alleged VPNs that are established only through an ISP. In particular, BinGO EFR describes two scenarios for establishing an alleged VPN: a point-to-point tunneling protocol (PPTP) Client-to-VPN Server scenario and a LAN-to-LAN VPN scenario. Appx2828-2829; Appx4106 (¶ 114). Under either scenario,

however, BinGO EFR describes that a connection is established to the local ISP first, and then an alleged VPN is established over the Internet. Appx2828-2829; Appx4106 (¶ 114).

Regarding authentication, BinGO UG explains that "[b]efore every connection, BinGO! and the router at HQ check the incoming data to see if they should take the call. In order to protect the network against unauthorized access, acceptance of the call only takes place after correct authentication. This authentication is based on a common password and two codes that you and your partner use for the connection." Appx2394.

IV. PROCEDURAL BACKGROUND.

A. The *Inter Partes* Reexamination Proceedings Below

Apple's request for *inter partes* reexamination advanced twenty four different grounds of unpatentability against claims 1-18 of the '135 patent based on various asserted references. Appx137-138. After Apple failed to prove invalidity in the district court litigation, the Patent Office terminated the reexamination with respect to claims 1-9 under 35 U.S.C. § 317(b) (2006)—the statutory estoppel provision governing *inter partes* reexaminations. Appx6788-6804. After a prolonged examination spanning over six years, the Examiner ultimately issued a Right of Appeal Notice ("RAN") directed to claims 10-18, adopting various rejections

against these claims. Appx6865-6869. VirnetX appealed the Examiner's findings to the Board. Appx6906-6977; Appx7005-7024; Appx7035-7037; Appx7038-7047.

B. The Board's Decision

The Board's decision identified various problems in the Examiner's RAN. For instance, the Examiner adopted rejections based on a reference referred to as Aventail that even Apple acknowledged as defective. Appx9-10. The Board also faulted the adoption of several rejections based on a reference referred to as Wang. Appx10-17.[2]

With respect to various rejections based on Beser, the Board agreed with VirnetX that most of the rejections could not stand. Appx22-24. As to claim 18, however, the Board found that some of VirnetX's arguments were precluded by collateral estoppel due to proceedings involving different VirnetX patents. Appx27-30. The Board also found that the authorization limitation—requiring that "prior to automatically initiating the VPN between the client computer and the target

[2] The Board did affirm the rejection of claims 13, 14, and 17 based on Wang, either alone or together with other references. Appx17-21. In another *inter partes* reexamination proceeding, No. 95/001,679, the Board found these claims unpatentable on the basis of a different reference, Kiuchi (which is not at issue in this appeal). An appeal from that reexamination proceeding is pending before this Court as No. 19-1671. In the interest of judicial efficiency, VirnetX did not challenge the Board's determination of unpatentability regarding claims 13-17 in No. 19-1671. As such, the Board's Wang-based rejections of claims 13, 14, and 17 below are effectively moot, and VirnetX is not challenging them here.

computer, determining whether the client computer is authorized to resolve addresses of non secure target computers and, if not so authorized, returning an error from the DNS request"—while not precluded by collateral estoppel, was taught by the combination of Beser, Kent, and Blum. Appx30-32. In making this finding, the Board asserted that "[t]he language of claim 18 is confusing because the language appears to be directed to the disclosed scenario at column 40, lines 4-13 of the '135 Patent, but the claim language does not clearly recite this scenario." Appx31. Relying on this professed confusion, the Board set aside the actual language of the limitation and instead proceeded to "decide whether as a general matter, the prior art would have rendered obvious determining whether a client has permission to access a web site and if not, returning an error message." Appx31. The Board concluded that the asserted references met this simplified version of claim 18's authorization limitation. Appx31-32.

The Board also affirmed the Examiner's anticipation rejection of claims 10, 12, 13, and 18 based on BinGO (as well as obviousness rejections of dependent claims 11 and 17). Appx32-49. The Board opined that it was proper to rely on both BinGO UG and BinGO EFR in finding anticipation because "BinGO UG expressly refers to BinGO EFR in several places for detailed information in implementing specific functions," and that, as a result, "BinGO UG incorporates by reference BinGO EFR." Appx35-36. With respect to the authorization limitation of claim 18,

the Board "reiterate[d] the issues with claim 18 discussed above," Appx45 (i.e., its purported confusion that led it to rewrite the authorization limitation, Appx31). After redefining claim 18 in this way, the Board reasoned that BinGO could anticipate claim 18 and teach the authorization limitation because "BinGO discloses authentication of [an] initiating partner, which would include situations where the requested web-site is non-secure," and asserted that "[a]lthough it is true that BinGO does not expressly disclose returning an error message should the authentication fail," it "d[id] not find it credible to argue that one of ordinary skill in the art would not have immediately understood that such an authentication failure would result in an error message being returned." Appx45.

C. The Board's Consideration of VirnetX's Rehearing Request

VirnetX sought rehearing of the Board's decision. Appx7051-7057. In addition to challenging the substance of the Board's unpatentability determinations, as part of the request VirnetX also pointed to the "imminent guidance from the Supreme Court regarding the proper remedy for decisions rendered by Board panels whose appointments did not comport with the requirements of the Appointments Clause." Appx7055. VirnetX explained how "Arthrex … argued to the Supreme Court that APJs remain unconstitutionally appointed principal officers" even following this Court's ruling in *Arthrex, Inc. v. Smith & Nephew, Inc.*, 941 F.3d 1320, 1335 (Fed. Cir. 2019), and thus requested that "whatever remedy is provided

by the Supreme Court should be provided in the present reexamination." Appx7055-7056.

After the Supreme Court's decision in *Arthrex* issued, the Board "referred VirnetX's request to Mr. Hirshfeld, Commissioner for Patents, Performing the Functions and Duties of the Under Secretary of Commerce for Intellectual Property and Director of the United States Patent and Trademark Office." Appx56. Commissioner Hirshfeld then issued an order denying VirnetX's rehearing request without explanation. Appx56.

SUMMARY OF ARGUMENT

The Board committed several errors that compel reversal, or at least vacatur, of the final written decision.

First, the Board expressly disregarded the actual language of claim 18, which recites that "prior to automatically initiating the VPN between the client computer and the target computer, determining whether the client computer is authorized to resolve addresses of non secure target computers and, if not so authorized, returning an error from the DNS request." Appx134 (claim 18). The Board asserted that "claim 18 is confusing because the language appears to be directed to the disclosed scenario at column 40, lines 4-13 of the '135 patent, but the claim language does not clearly recite this scenario." Appx31. As a result, the Board elected to only "decide whether as a general matter, the prior art would have rendered obvious determining

whether a client has permission to access a web site and if not, returning an error message." Appx31. The Board's approach, which negated several limitations of claim 18, violated the "bedrock principle of patent law that the claims of a patent define the invention to which the patentee is entitled the right to exclude." *Phillips v. AWH Corp.*, 415 F.3d 1303, 1312 (Fed. Cir. 2005) (en banc) (internal quotation marks and citations omitted).

Regardless, the Board was wrong about the authorization limitation's relationship to the specification. The limitation at issue in claim 18—one that the Board professed not to see in the claims—is, in fact, found in the original claims filed with the application that led to the '135 patent, and "[o]riginal claims are part of the specification." *Crown Packaging Tech.*, 635 F.3d at 1380. With the Board's misreading of claim 18's limitations corrected, neither the Beser-based rejections nor the BinGO-based rejections satisfy the actual language of the claim.

Second, in addressing other limitations of claim 18, the Board applied collateral estoppel to certain of VirnetX's arguments regarding the Beser-based rejections. One of VirnetX's arguments, however, was that the combination of Beser, Kent, and Blum did not teach "determining whether the DNS request transmitted in step (1) is requesting access to a secure web site," as recited in claim 18. Appx6964-6965; *see also* Appx6957-6958. None of the past proceedings identified by the Board involved this or similar claim language, or the issues

underlying VirnetX's argument. As such, application of collateral estoppel was improper.

Third, the Board improperly relied on the combination of multiple references, BinGO UG and BinGO EFR, in finding anticipation. Noting that "BinGO UG expressly refers to BinGO EFR in several places for detailed information in implementing specific functions," the Board concluded that "BinGO UG incorporates by reference BinGO EFR." Appx35 (citing Appx2480; Appx2591; Appx2631; Appx2644). That was error. "To incorporate material by reference, the host document must identify with detailed particularity what specific material it incorporates and clearly indicate where that material is found in the various documents." *Advanced Display Sys., Inc. v. Kent State Univ.*, 212 F.3d 1272, 1282 (Fed. Cir. 2000); *see also In re de Seversky*, 474 F.2d at 674 ("a mere *reference* to another application, or patent, or publication is not an *incorporation* of anything therein"). That strict standard was not met here.

Fourth, the Board failed to comply with its own regulations when it referred VirnetX's request for rehearing to the Commissioner for Patents, who was then performing the functions of the Patent Office's Director, rather than considering the merits of the rehearing request by a three-judge panel. The Board's regulations prescribe that, "[i]f a party to an appeal files a request for rehearing under paragraph (a) of this section … the Board *shall* render a decision on the request for rehearing."

37 C.F.R. § 41.79(d) (emphasis added). By referring VirnetX's request instead to the Commissioner for Patents, who then issued a perfunctory rejection of VirnetX's request, the Board did not render a decision on VirnetX's request. The Board's decision to refer the request for *Director* review instead of setting it for panel rehearing is particularly faulty given the Board's stated view that Director review is not available for *inter partes* reexaminations.

ARGUMENT

I. STANDARD OF REVIEW

This Court reviews the Board's legal conclusions de novo and its factual findings for substantial evidence. *EmeraChem Holdings, LLC v. Volkswagen Grp. of Am., Inc.*, 859 F.3d 1341, 1345 (Fed. Cir. 2017). Anticipation is a fact question reviewed for substantial evidence. *Power Integrations, Inc. v. Lee*, 797 F.3d 1318, 1323 (Fed. Cir. 2015). Factual findings underlying obviousness are also reviewed for substantial evidence, while the ultimate question of obviousness is reviewed de novo. *Dell Inc. v. Acceleron, LLC*, 818 F.3d 1293, 1298 (Fed. Cir. 2016). Whether and to what extent material has been incorporated by reference into a host document is a question of law. *Advanced Display Sys.*, 212 F.3d at 1283. This Court reviews de novo questions of statutory or constitutional interpretation. *U.S. Shoe Corp. v. United States*, 296 F.3d 1378, 1381 (Fed. Cir. 2002).

II. THE BOARD'S REJECTIONS BASED ON BESER, KENT, AND BLUM ARE FUNDAMENTALLY FLAWED

A. The Board Improperly Recrafted Claim 18 and Failed to Address Its Limitations

Claim 18 requires, among other things, a particular authorization mechanism. Specifically, claim 18 requires that "prior to automatically initiating the VPN between the client computer and the target computer, determining whether the client computer is authorized to resolve addresses of non secure target computers and, if not so authorized, returning an error from the DNS request." Appx134 (2:16-21). The Board acknowledged this language, Appx30, but then asserted that "claim 18 is confusing because the language appears to be directed to the disclosed scenario at column 40, lines 4-13 of the '135 patent, but the claim language does not clearly recite this scenario." Appx31. As a result, instead of addressing the claim as written, the Board elected to "decide whether as a general matter, the prior art would have rendered obvious determining whether a client has permission to access a web site and if not, returning an error message." Appx31.

The Board's redrafting of claim 18 omits key limitations. In particular, (1) the Board erased the requirement that the determination be performed "prior to automatically initiating the VPN between the client computer and the target computer"; (2) the Board changed "determining whether the client computer is authorized to resolve addresses of non secure target computers," as recited in claim

18, to simply "determining whether a client has permission to access a web site"; and (3) the Board changed "returning an error from the DNS request," as recited in claim 18, to simply "returning an error message." The Board's approach of ignoring the actual claim language in favor of its own paraphrasing was patently erroneous.

"It is a bedrock principle of patent law that the claims of a patent define the invention to which the patentee is entitled the right to exclude." *Phillips*, 415 F.3d at 1312 (internal quotation marks and citations omitted). "It is the claims that define the metes and bounds of the patentee's invention. The claims, not specification embodiments, define the scope of patent protection." *Kara Tech.*, 582 F.3d at 1347-48 (citation omitted). The Board's apparent attempt to remove limitations from the claim that it did not find in one particular example of the specification defies this Court's precedent, even if the Board was correct that "the claim language [of claim 18] does not clearly recite th[e] scenario [at column 40, lines 4-13 of the '135 patent]." Appx31.

The Board's premise that the recited portion of claim 18—"prior to automatically initiating the VPN between the client computer and the target computer, determining whether the client computer is authorized to resolve addresses of non secure target computers and, if not so authorized, returning an error from the DNS request"—lacked support in the specification was erroneous in any event. That language is supported by the original claims, which are part of the

specification. As this Court has explained, "[o]riginal claims are part of the specification and in many cases will satisfy the written description requirement." *Crown Packaging Tech.*, 635 F.3d at 1380. For instance, original claim 32, filed with the original application that ultimately issued as the '135 patent, recited precisely the same limitation (and ultimately issued as claim 5 of the '135 patent). *See* Application No. 09/504,783, Claims filed February 15, 2000, at 4-5, available at https://patentcenter.uspto.gov/applications/09504783/ifw/docs (find the document dated "02/15/2000" with the document description "Claims," and then click "PDF"); *see also* Application/Control No. 95/001,269, Remarks filed April 15, 2010, at 1, available at https://patentcenter.uspto.gov/applications/95001269/ifw/docs (find the document dated "04/15/2010" with the document description "Applicant Arguments/Remarks Made in an Amendment," and then click "PDF") (identifying issued claim 5, which is original claim 32, as providing support for claim 18).

The Board's decision to evaluate obviousness based on its paraphrasing of claim 18, rather than claim 18's actual text, was outcome determinative. The Board had rejected the Examiner's analysis regarding claim 18 and the Beser and Kent references. It observed that "the rejection of claim 18 as set forth in the Request and adopted by the Examiner does not explain how claim 18 would have been obvious over Beser and Kent alone, but rather relies on Blum for the client authorization recitation." Appx31; Appx310-312. The Board found obviousness only by

reformulating the claims to exclude the missing limitations. It pointed to Beser and Kent's generic authorization-related disclosures: Beser's disclosure that "IP packets may require authentication," Appx32 (citing Appx2260 (11:22-25)), and Kent's disclosure of "access control, which is 'prevention of use of a resource in an unauthorized manner,'" Appx32 (citing Appx2275; Appx2316). In addition, the Board asserted that Blum "discloses employing protocol filters in order to impart specific capabilities and functions, and when services are not available to a client computer, an error message may be returned to the client." Appx32 (citing Appx2351 (7:35-38, 7:56-58, 8:65-9:3)).

There were multiple problems with the Board's assertions. For one, the Board's characterization of Blum was not accurate; with respect to the error message, Blum discloses only that "when none of the DNS services available to the client compute system 300 are able to resolve the requested host name to an IP address, an error message is returned." Appx2351-2352 (8:65-9:3). Regardless, the Board found only that "the determination of what web sites, whether secure or non-secure, are available to the client would have been an obvious variation of the authentication procedures disclosed in Beser, Kent, and Blum, such that determining whether a client is authorized to access a non secure target computer and returning an error if the client is not authorized to access a non-secure target as recited in claim 18 would have been obvious." Appx32. Thus, the Board did not even

consider, much less determine, whether it would have been obvious to "determin[e] whether the client computer is authorized to resolve addresses of non secure target computers and, if not so authorized, return[] an error from the DNS request," as claimed, or to perform such authorization "prior to automatically initiating [a] VPN," as also claimed. Appx134 (claim 18). Nor could the Board make such determination given its stated intention to rely on the generic prior art disclosures that do not meet the express language in the claim.

The Board's obviousness analysis itself was also defective in that it failed to properly analyze the motivation to combine Beser, Kent, and Blum. Even under the Board's rewriting of the claim, but particularly given the narrow requirements of the claim, the Board's analysis was cursory, and only pointed to a few disclosures of the prior art in isolation. That is insufficient. "[T]he factual inquiry whether to combine references must be thorough and searching, and the need for specificity pervades [this Court's] authority on the PTAB's findings on motivation to combine." *In re NuVasive, Inc.*, 842 F.3d 1376, 1381-82 (Fed. Cir. 2016) (internal quotation marks and citation omitted). There was no such specificity here. For this additional reason, the Board's finding of obviousness should be set aside.

B. The Board Failed to Address VirnetX's Arguments as to the Remaining Limitations Based on an Erroneous Application of Collateral Estoppel

In addition to the determining step discussed above, VirnetX challenged whether the combination of Beser, Kent, and Blum taught several other limitations of claim 18, including "determining whether the DNS request transmitted in step (1) is requesting access to a secure web site." Appx6964-6965; *see also* Appx6957-6958. The Board, however, failed to properly address this argument.

The Board asserted that VirnetX was "collaterally estopped from relying on such arguments." Appx27 (referring to VirnetX's "argument[] that Beser does not render obvious … determining whether the request is requesting access to a secure/non-secure web site"); *see also* Appx29 (opining that "the issue[] of whether as a general matter … Beser determines that a requested web-site is secure/non-secure [is] identical and actually litigated in the aforementioned *inter pates* review proceedings"). The Board's finding of collateral estoppel fails.

As an initial matter, VirnetX was denied a fair opportunity to respond to this argument, which was never presented by the Examiner or Apple. While the Board indicated that Apple "observe[d] … that several issues identified by Patent Owner with respect to the combination of Beser and Kent have been decided previously by the Board in other proceedings," Appx27, Apple in fact never argued collateral estoppel. *See, e.g.*, Appx7020-7021; *see also* Appx7008 (identifying other

proceedings only as generally presenting "issues that are the same or similar to issues present in this appeal"). Since VirnetX did not have a "'fair opportunity to react to the thrust of the rejection,'" the Board's collateral estoppel finding should be set aside. *See In re Biedermann*, 733 F.3d 329, 337 (Fed. Cir. 2013) (quoting *In re Leithem*, 661 F.3d 1316, 1319 (Fed Cir. 2011)); *see also In re NuVasive, Inc.*, 841 F.3d 966, 972 (Fed. Cir. 2016).

In any event, there is no basis to apply collateral estoppel. In addressing past Board findings relating to Beser (which supposedly served as grounds for collateral estoppel), the Board observed only that VirnetX "has argued in multiple inter partes review proceedings in which the combination of Beser and Kent have been applied to claims reciting a request for an IP address based on a domain name and where a virtual private network link is established, that Beser does not disclose or suggest DNS look up functionality and that Beser teaches away from the use of IP sec protocol in Kent, arguments that were rejected by the Board and where the Decisions were affirmed by the Federal Circuit." Appx28-29. But the question in the proceeding below was more specific—it was whether the combination of Beser, Kent, and Blum teaches determining whether a DNS request is requesting access to a secure web site. The Board thus did not find that any of the prior proceedings concerned this precise

issue. Nor could the Board have found so, given the markedly different claim language at issue in the other proceedings identified by the Board.[3]

III. THE BOARD'S REJECTIONS BASED ON BINGO ARE FUNDAMENTALLY FLAWED

A. The Board Improperly Recrafted Claim 18 and Failed to Address Claim 18's Limitations

As discussed above, claim 18 requires, among other things, a particular authorization mechanism: "prior to automatically initiating the VPN between the client computer and the target computer, determining whether the client computer is authorized to resolve addresses of non secure target computers and, if not so authorized, returning an error from the DNS request." Appx134 (claim 18); *see also supra* at 23. In addressing this limitation, the Board reiterated its misguided approach that ignored the actual claim language on the grounds that the Board could not find the same limitation in the specification. *See* Appx45 ("As to determining whether the client computer is authorized to access non-secure target computers and returning an error message if it determined the client is not authorized to access non-secure target computers, we reiterate the issues with claim 18 discussed above."). As a result, the Board analyzed only its own rewriting of the determining limitation,

[3] The patents at issue in the proceedings identified by the Board are U.S. Patent Nos. 8,504,696, 8,868,705, 8,850,009, 8,458,341, 8,516,131, and 8,560,705. None of them contains a limitation directed to determining whether a DNS request is requesting access to a secure web site.

and found that "BinGO discloses authentication of the initiating partner, which would include situations where the requested web-site is non-secure" and that "authentication failure would result in an error message being returned." Appx45. For reasons discussed above, *see supra* at 23-27, the Board's recrafting of claim 18 was improper.

The Board committed an additional error in its analysis of the determining step. Not only did the Board fail to address the limitation as written, the Board did not even properly find that BinGO taught the Board's simplified version of the limitation. Specifically, the Board acknowledged that "it is true that BinGO does not expressly disclose returning an error message should authentication fail," but did "not find it credible to argue that one of ordinary skill in the art would not have immediately understood that such an authentication failure would result in an error message being returned." Appx45. This finding lacks any evidence whatsoever.

For starters, the asserted ground of unpatentability based on BinGO was anticipation, not obviousness. Appx32. Under an anticipation analysis, "a reference missing a limitation cannot anticipate even if a skilled artisan would 'at once envisage' the missing limitation." *Genentech, Inc. v. Hospira, Inc.*, 946 F.3d 1333, 1340 (Fed. Cir. 2020) (quoting *Nidec Motor Corp. v. Zhongshan Broad Ocean Motor Co. Ltd*, 851 F.3d 1270, 1274-75 (Fed. Cir. 2017)). The Board's concession

that BinGO does not expressly recite returning an error message is fatal to any theory of anticipation.

In addition, as VirnetX's expert, Dr. Angelos Keromytis, explained, BinGO operates differently than claim 18. It simply describes "authenticating against unauthorized access to the network," but "determining if the user's PC is authorized to access a network is not the same as determining if the user's PC is authorized to resolve addresses not located at that network." Appx4122 (¶ 163). As Dr. Keromytis further explained, BinGO says nothing about performing such a determination with respect to "non secure target computers," Appx4123 (¶ 164), let alone performing the claimed determination "prior to [when] the alleged VPN between the user's PC and the BossPC is initiated," Appx4123 (¶ 165). And BinGO also fails to say anything about "returning an error from the DNS request if the client computer is not authorized to resolve addresses of non secure target computers." Appx4123-4124 (¶ 166). There is simply no evidence to support the Board's conclusion that "one of ordinary skill in the art would … have immediately understood that … an authentication failure [in BinGO] would result in an error message being returned," even if such was sufficient to meet the claim (it is not) or was a finding that could support a finding of anticipation (it cannot).

B. The Board Improperly Relied on a Combination of Multiple References for Anticipation

The anticipation rejections of claims 10, 12, and 18 based on BinGO are also defective because they improperly rely on the combination of two different rejections, referred to as "BinGO UG" and "BinGO EFR." *See, e.g.*, Appx324-332; Appx340-342; Appx347-352. Apple misleadingly packaged BinGO UG and BinGO EFR together as a single exhibit, but they are indisputable different documents published at different times. *See, e.g.*, Appx35-36 (recognizing that BinGO UG bears a March 1999 date, while BinGO EFR bears a February 1999 date); *see also* Appx2354-2366; Appx2738-2744 (showing separate dates, separate tables of contents, and separate page numbering). The Board nonetheless relied on disclosures from both BinGO UG and BinGO EFR to support a single anticipation rejection on the specious rationale that "BinGO UG expressly refers to BinGO EFR in several places for detailed information in implementing specific functions." Appx35 (citing Appx2480; Appx2591; Appx2631; Appx2644). The Board took these references to mean that "BinGO UG incorporates by reference BinGO EFR." Appx35. The Board erred as a matter of law.

As this Court has explained, "[t]o incorporate material by reference, the host document must identify with detailed particularity what specific material it incorporates and clearly indicate where that material is found in the various documents." *Advanced Display Sys.*, 212 F.3d at 1282. The Board did not even

attempt to make this requisite finding. Nowhere did the Board explain what "specific material" BinGO UG incorporates from BinGO EFR, nor did the Board explain "where that material is found" in BinGO EFR.

Nor could the Board have properly made such findings here. The Board identified four pages of BinGO UG in support of its incorporation-by-reference finding: pages 115, 226, 266, and 279. Appx35. Page 115 indicates, when discussing "necessary settings for Virtual Private Networking (VPN)," that "[y]ou will find more detailed explanations and instructions in Extended Features Reference." Appx2480. Similarly, pages 226 and 266 of BinGO UG indicate, again when discussing settings for virtual private networking, that "You can find detailed information and configuration instructions (with examples) in Extended Feature Reference." Appx2591; Appx2631. But "a mere *reference* to another application, or patent, or publication is not an *incorporation* of anything therein." *In re de Seversky*, 474 F.2d at 674; *see also In re Saunders*, 444 F.2d 599, 602-03 (C.C.P.A. 1971) (a rejection for anticipation is appropriate only if one reference "expressly incorporates a particular part" of another reference). BinGO UG expresses no intent to *incorporate* anything in BinGO EFR, much less something specific. Nor does BinGO UG indicate where any such specific item could be found in BinGO EFR; instead, it simply directs the reader to BinGO EFR if the reader wants to "find more detailed explanations and instructions," Appx2480, or to "find detailed information

and configuration instructions," Appx2591; Appx2631. And page 279 of BinGO UG simply indicates that some "general product information" can be found in several documents, including "Software Reference, Extended Features Reference, [and] BRICKware for Windows." Appx2644. Again, there is no clear indication that BinGO UG even intends to incorporate by reference anything in BinGO EFR, much less does it "identify with detailed particularity what specific material it incorporates and clearly indicate where that material is found." *Advanced Display*, 212 F.3d at 1282.

The references to BinGO EFR contained in BinGO UG are at best akin to a citation, which is not sufficient for an incorporation by reference. *See Commonwealth Sci. & Indus. Rsch. Organisation v. Buffalo Tech. (USA), Inc.*, 542 F.3d 1363, 1372 (Fed. Cir. 2008) ("The footnote citation … could provide a justification for combining the references for obviousness purposes, but there is nothing about the reference to Bingham that appears to constitute an incorporation of any or all of the information from the Bingham reference under the standard set forth in *Advanced Display Systems.*"). This Court's discussion of the GSM standard in *Kyocera Wireless Corp. v. International Trade Commission*, 545 F.3d 1340 (Fed. Cir. 2008), is instructive. As the Court explained, "[t]he GSM standard is a comprehensive set of specifications for a second generation ('2G') mobile network." *Id.* at 1350. Each specification that is part of the GSM standard has "its own title

and separate page numbering." *Id.* at 1351. Nonetheless, appellant "assert[ed] that each GSM specification incorporates the others by reference." *Id.* at 1351. As the Court explained, however, even though "specifications at times cross-reference other specifications," "[t]his vague referencing practice is hardly sufficient to meet this court's legal requirements for incorporation." *Id.* at 1352.

Finally, the Board's obviousness rejection of claim 11 based on BinGO also fails for the same reason. While obviousness permits the combination of multiple references, here the Board did not make the requisite findings under *Graham v. John Deere Co*., 383 U.S. 1 (1966), to combine BinGO UG and BinGO EFR. To the contrary, the Board continued to treat both references as a single BinGO reference. Appx45-47; Appx49.

IV. THE BOARD ERRONEOUSLY REFUSED TO PROPERLY CONSIDER VIRNETX'S REQUEST FOR REHEARING

A. The Board's Treatment of VirnetX's Rehearing Request Did Not Comply with the Applicable Regulations

At a minimum, vacatur and remand is required because the Board refused to consider properly VirnetX's request for rehearing of the Board's May 14, 2021 final written decision. Appx7051. VirnetX's request for rehearing was made under the Board's regulation in 37 C.F.R. § 41.79, which grants "[p]arties to [an] appeal" the right to "file a request for rehearing of [a] decision within one month." 37 C.F.R. § 41.79(a); *see also id.* § 42.71(d) ("A party dissatisfied with a decision may file a

single request for rehearing without prior authorization from the Board."). The rule prescribes that, "[i]f a party to an appeal files a request for rehearing under paragraph (a) of this section … the Board *shall* render a decision on the request for rehearing." 37 C.F.R. § 41.79(d) (emphasis added).

VirnetX's rehearing request identified a host of problems with the Board's May 14, 2021 decision, including the Board's treatment of expert evidence, Appx7052, and its consideration of the asserted references, Appx7052-7055. In addition, because the Supreme Court was poised to render its decision regarding the proper remedy for the Appointments Clause violations this Court had identified in *Arthrex, Inc. v. Smith & Nephew, Inc.*, 941 F.3d 1320 (Fed. Cir. 2019), VirnetX also requested "whatever remedy is provided by the Supreme Court." Appx7055-7056.

Rather than substantively consider VirnetX's rehearing request, the Board "referred VirnetX's request to Mr. Hirshfeld, Commissioner for Patents, Performing the Functions and Duties of the Under Secretary of Commerce for Intellectual Property and Director of the United States Patent and Trademark Office." Appx56. The Board notionally did so in light of VirnetX's request for "whatever remedy is provided by the Supreme Court" in *Arthrex*, and the Supreme Court's subsequent decision. *Id.* Commissioner Hirshfeld then summarily denied VirnetX's request without explanation. Appx56.

The Board's approach defied the regulation governing rehearing requests in *inter partes* reexaminations, which obligates the Board to issue a decision when a timely request for rehearing is filed. *See* 37 C.F.R. § 41.79(d) ("the Board *shall* render a decision on the request for rehearing") (emphasis added). Commissioner Hirshfeld's summary rejection of VirnetX's request does not qualify as such a decision in conformance with the requirements of the Administrative Procedure Act ("APA"). The APA obligates the Board "to come to a sound decision" and "to fully and particularly set out the bases upon which it reached that decision." *Power Integrations, Inc. v. Lee*, 797 F.3d 1318, 1323 (Fed. Cir. 2015); *see also Pers. Web Techs., LLC v. Apple, Inc.*, 848 F.3d 987, 992 (Fed. Cir. 2017) ("The Board, as an administrative agency, 'must articulate "logical and rational" reasons for [its] decision.'") (quoting *Synopsys, Inc. v. Mentor Graphics Corp.*, 814 F.3d 1309, 1322 (Fed. Cir. 2016)). The Board thus acts improperly when it "fails to adequately evaluate" a party's arguments. *Power Integrations*, 797 F.3d at 1325; *see also In re Hodges*, 882 F.3d 1107, 1116 (Fed. Cir. 2018) ("The Board must … 'explicate its factual conclusions, enabling us to verify readily whether those conclusions are indeed supported by "substantial evidence" contained within the record.'") (quoting *In re Gartside*, 203 F.3d 1305, 1314 (Fed. Cir. 2000)); *Shinn Fu Co. of Am., Inc. v. Tire Hanger Corp.*, 701 F. App'x 942, 946 (Fed. Cir. 2017) ("Because the Board did not provide any analysis with regard to the manner in which Shinn Fun proposed its

key obviousness combination, we have no meaningful way to review the Board's patentability determination in light of Shinn Fu's arguments."). Commissioner Hirshfeld's denial of VirnetX's request provides no indication whatsoever as to why VirnetX's request was rejected.

This is not to suggest that the Director necessarily must present a detailed analysis any time a request for Director review is submitted. Here, however, VirnetX did *not* submit a request for Director review. VirnetX's rehearing request was filed under the provisions of the regulation governing ordinary rehearing request in *inter partes* reexaminations, 37 C.F.R. § 41.79. *See* Appx7051. The Board's decision to recast VirnetX's request as a request for Director review, Appx56, was improper, at least insomuch as the Board did so as a *substitute* for rehearing under 37 C.F.R. § 41.79. VirnetX's request for "whatever remedy is provided by the Supreme Court [in *Arthrex*]," Appx7055-7056, cannot properly be construed as a request for such relief *instead of* the rehearing to which VirnetX was entitled under section 41.79, as the Supreme Court never held that a party may only request *either* panel rehearing *or* Director review—that is an invention of the Patent Office's own creation. *See, e.g.*, U.S. Patent and Trademark Office, Interim Process for Director Review, available at https://www.uspto.gov/patents/patent-trial-and-appeal-board/interim-process-director-review ("Under the interim process, parties are limited to requesting either Director review or a rehearing by the original Board

panel. … Requests for both Director review and panel rehearing of the same decision are treated as a request for Director review only.").

The Supreme Court in *Arthrex* prescribed a "tailored" remedy to the Appointments Clause violation: It held 35 U.S.C. § 6(c)—the statutory provision that required rehearing "by at least 3 members of the … Board"—"unenforceable *as applied to the Director* insofar as it prevents the Director from reviewing the decisions of the PTAB on his own." *United States v. Arthrex, Inc.*, 141 S. Ct. 1970, 1987 (2021) (emphasis added). The Supreme Court did not make the panel rehearing option (to which a party is entitled under the Board's regulations) and the Director review option interchangeable. Even if Director review alone might be sufficient under the statute governing *inter partes* reviews and the Appointments Clause, neither *requires* that it be the only review mechanism provided by the Board, and the plain terms of 37 C.F.R. § 41.79 entitle a party to another mechanism—namely, a panel rehearing. As long as that still-effective regulation remains on the books, the Board has to follow it. *See Wagner v. United States*, 365 F.3d 1358, 1361 (Fed. Cir. 2004) ("an agency is bound by its own regulations") (citing *Service v. Dulles*, 354 U.S. 363, 388 (1957)). And, to be sure, the Board never asked VirnetX whether it wanted to convert its request for rehearing into a request for Director review.

There is no indication that Commissioner Hirshfeld even substantively considered VirnetX's rehearing request. The Patent Office maintains a list of

completed requests for Director reviews. *See* U.S. Patent and Trademark Office, Status of Director Review Requests, available at https://www.uspto.gov/patents/patent-trial-and-appeal-board/status-director-review-requests (follow "Director review requests spreadsheet"). Previously, as part of that list, the Patent Office provided "issue(s)" associated with each request. In the entry for the request in this case, the first issue was "[w]hether the Supreme Court's holding in Arthrex applies to inter partes reexams"—implying that, in the Patent Office's view, there is a question of whether Director review even applies in *inter partes* reexaminations. *See* U.S. Patent and Trademark Office, Status of Director Review Requests, archived on July 29, 2022, available at https://web.archive.org/web/20220729000429/https://www.uspto.gov/patents/patent-trial-and-appeal-board/status-director-review-requests (follow "Director review requests spreadsheet," click on "Completed" tab, and see row 2, identifying PTAB Case Number "2021-001315 (95/001,682)").[4] This was consistent with Apple's argument below that "*Arthrex* does not apply to *inter partes* reexamination proceedings such as this one." Appx7061-7062. It is thus entirely possible that

[4] In the August 2022 version of the spreadsheet available at the time this brief is being submitted, it appears the Patent Office no longer provides such "issue" information to the public. As such, VirnetX is citing the Internet Archive, documents the July version of the spreadsheet.

Commissioner Hirshfeld found that Director review was not even available for *inter partes* reexaminations, and denied VirnetX's request for that reason.

Indeed, evidence suggests that is exactly what happened. In a co-pending *inter partes* reexamination between VirnetX and Apple, where VirnetX explicitly requested Director review, the Board indicated that "'parties may only request Director review of final written decisions issued in inter partes reviews and post-grant reviews'" and, as a result, "[VirnetX] cannot request Director review in this [*inter partes* reexamination] proceeding." Application/Control No. 95/001,697, Decision entered May 6, 2022, at 2, available at https://patentcenter.uspto.gov/applications/95001697/ifw/docs (find the document dated "05/06/2022" with the document description "Board of Appeals Decision on Rehearing," and then click "PDF") (quoting U.S. Patent and Trademark Office, *Arthrex* Q&A, A9, available at https://www.uspto.gov/patents/patent-trial-and-appeal-board/procedures/arthrex-qas).[5] It thus appears highly likely that the Board refused to consider VirnetX's request for rehearing in this *inter partes* reexamination by unilaterally converting it into a request for Director review—which was then denied on the ground that Director review is not available in *inter partes* reexaminations.

[5] An appeal from the Board's decision in Application/Control No. 95/001,697 is currently pending before this Court as No. 22-1997.

That position, however, is contrary to this Court's precedent. As the Court made clear, the holding in *Arthrex* extends to *inter partes* reexaminations. *See, e.g.*, *VirnetX Inc. v. Cisco Sys., Inc.*, 958 F.3d 1333, 1336-37 (Fed. Cir. 2020); *see also VirnetX Inc. v. Cisco Sys., Inc.*, No. 19-1671, Dkt. No. 63 (Fed. Cir. Oct. 5, 2021) (applying the Supreme Court's remedy in *Arthrex* to an *inter partes* reexamination). At a minimum, the Court should vacate the decision below and remand to the Board to conduct a proper rehearing based on VirnetX's request under 37 C.F.R. § 41.79.

B. The Appointments Clause, the Supreme Court's *Arthrex* Decision, and the FVRA Require a Remand to a Permanent PTO Director

To the extent VirnetX's rehearing request could properly be referred to the Patent Office's Director, the Appointments Clause, the Supreme Court's *Arthrex* decision, and the Federal Vacancies Reform Act of 1998 ("FVRA") require a presidentially appointed, Senate-confirmed principal officer to make a final decision for the Board. Commissioner Hirshfeld, however, was appointed as an inferior officer by the Secretary of Commerce—just like the other Board members who lack the power under the Constitution to issue final decisions for the Executive Branch. *Arthrex*, 141 S. Ct. at 1987. Permitting an inferior officer to have the Executive's final say contravenes *Arthrex*'s express holding that "*[o]nly* an officer properly *appointed to a principal office* may issue a final decision." *Id.* at 1985 (emphasis added).

The FVRA independently barred Commissioner Hirshfeld from exercising the Director's exclusive authority to issue final decisions on rehearing. When a principal office is vacant, any functions or duties "required by statute to be performed by the [principal] officer (and only that officer)" may be performed *only* by (1) the officer's "first assistant" or (2) someone directed by "the President (and only the President)" to perform them. 5 U.S.C. §§ 3345(a), 3348(a)-(b). Under *Arthrex*, singlehanded review of Board decisions is a function or duty *only* the Director may perform. Commissioner Hirshfeld was not the Deputy Director (i.e., "first assistant"), and he has not been directed by the President to perform the Director's functions and duties. Under the FVRA, Commissioner Hirshfeld's denial of rehearing is "of no force and effect." 5 U.S.C. § 3348(d)(1).

VirnetX acknowledges that its Appointments Clause and FVRA challenges to Commissioner Hirshfeld's authority in this Section IV.B are foreclosed by this Court's recent decision in *Arthrex, Inc. v. Smith & Nephew, Inc.*, 35 F.4th 1328 (Fed. Cir. 2022). VirnetX raises the arguments here to preserve them for potential en banc or Supreme Court review.

CONCLUSION

The Board's final written decision should be reversed or, at a minimum, vacated and remanded.

August 22, 2022

Respectfully submitted,

/s/ Naveen Modi
Naveen Modi
Stephen B. Kinnaird
Joseph E. Palys
Igor V. Timofeyev
Daniel Zeilberger
PAUL HASTINGS LLP
2050 M Street, N.W.
Washington, D.C. 20036
(202) 551-1700
(202) 551-1705 (fax)
naveenmodi@paulhastings.com

Counsel for Appellant VirnetX Inc.

ADDENDUM

ADDENDUM TABLE OF CONTENTS

UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
95/001,682	07/11/2011	6,502,135	077580-0132	1074

137313 7590 05/14/2021

PAUL HASTINGS LLP
2050 M. Street NW
Washington, DC 20036

EXAMINER	
PEIKARI, BEHZAD	

ART UNIT	PAPER NUMBER
3992	

MAIL DATE	DELIVERY MODE
05/14/2021	PAPER

Please find below and/or attached an Office communication concerning this application or proceeding.

The time period for reply, if any, is set in the attached communication.

PTOL-90A (Rev. 04/07)

UNITED STATES PATENT AND TRADEMARK OFFICE

BEFORE THE PATENT TRIAL AND APPEAL BOARD

APPLE INC.
Requester

v.

Patent of VIRNETX INC.
Patent Owner and Appellant

Appeal 2021-001315
Reexamination Control 95/001,682
Patent 6,502,135 B1
Technology Center 3900

Before KARL D. EASTHOM, JEFFREY B. ROBERTSON, and JEREMY J. CURCURI, *Administrative Patent Judges.*

ROBERTSON, *Administrative Patent Judge*.

DECISION ON APPEAL

I. STATEMENT OF THE CASE

VirnetX Inc. ("Patent Owner") appeals under 35 U.S.C. §§ 134(b) and 315(a) (Pre-AIA) from the Examiner's decision to reject claims 10–18.[1] Third-Party Requester Apple Inc. (hereinafter "Requester") urges that the Examiner's decision must be affirmed.[2] We have jurisdiction under 35 U.S.C. §§ 134(b) and 315(a) (Pre-AIA).

We affirm in part.

II. INTRODUCTION

A. Background and Summary

United States Patent 6,502,135 B1 (hereinafter the "'135 Patent"), which is the subject of the current *inter partes* reexamination, issued to Munger et al. on December 31, 2002. *Inter partes* reexamination was requested by Apple. Inc. ("REQUEST FOR INTER PARTES REEXAMINATION" filed on July 10, 2011, "Request"). Both Patent Owner and Requester identify numerous related appeals and proceedings. *See* Appeal Br. iv–vii; Resp't Br. iv, 1–2.

In particular, the Federal Circuit, in *VirnetX v. Mangrove Partners Master Fund, Ltd.*, 778 F. App'x 897 (Fed. Cir. 2019), vacated and remanded a PTAB Final Written Decision in IPR2015-01046, an *inter partes* review of the '135 Patent. Subsequently, the panel in that case issued a

---

[1] *See* Patent Owner's Appeal Brief filed February 6, 2020 (hereinafter "Appeal Br."); Examiner's Answer (mailed August 25, 2020) (hereinafter "Ans."); Right of Appeal Notice (mailed November 6, 2019) (hereinafter "RAN"); Patent Owner's Rebuttal Brief filed September 24, 2020 (hereinafter "Reb. Br.").

[2] *See* Respondent Brief (filed April 17, 2020) (hereinafter "Resp't Br.").

Final Written Decision on Remand Determining All Challenged Claims Unpatentable (claims 1, 3, 4, 7, 8, 10, 12). *Mangrove Partners Master Fund, LTD v. VirnetX*, IPR2015-01046, Paper No. 112 (PTAB July 14, 2020).

In addition, the '135 Patent is the subject of *inter partes* reexamination control no. 95/001,679 requested by Cisco Systems, Inc., and in which a Decision on Appeal (Appeal No. 2017-011862) was entered on February 6, 2018, which affirmed the Examiner's rejections of claims 1–9 and 13–18.[3]

B. The '135 Patent

The '135 Patent describes a system and method for communicating over the Internet and the automatic creation of a virtual private network (VPN) in response to a domain-name server look-up function. *See* '135 Patent, col. 2, l. 66 – col. 3, l. 2, col. 37, ll. 19–21.

Claims 10, 13, and 18, the independent claims on appeal, are illustrative of the appealed subject matter, and read as follows:

> 10. A system that transparently creates a virtual private network (VPN) between a client computer and a secure target computer, comprising:

[3] Rehearing was requested by Patent Owner, and denied in a Decision mailed January 18, 2019. The Board's decisions were appealed to the Federal Circuit. A General Order by the Chief Administrative Patent Judge entered on May 5, 2020 in the 95/001,679 reexamination indicated that the Federal Circuit had issued orders vacating a number of decisions by the PTAB including the Board's Decision in 95/001,679 in view of *Arthrex, Inc. v. Smith & Nephew, Inc.*, 941 F.3d 1320 (Fed. Cir. 2019)(*cert. granted sub nom. United States v. Arthrex, Inc.*, 2020 WL 6037206 (Oct. 13, 2020)). The General Order ordered 95/001,679 be held in abeyance until the Supreme Court ultimately resolves the issues raised in *Arthrex*.

a DNS proxy server that receives a request from the client computer to look up an IP address for a domain name, wherein the DNS proxy server returns the IP address for the requested domain name if it is determined that access to a non-secure web site has been requested, and wherein the DNS proxy server generates a request to create the VPN between the client computer and the secure target computer if it is determined that access to a secure web site has been requested; and

a gatekeeper computer that allocates resources for the VPN between the client computer and the secure web computer in response to the request by the DNS proxy server.

13. A method of establishing communication between one of a plurality of client computers and a central computer that maintains a plurality of authentication tables each corresponding to one of the client computers, the method comprising the steps of:

(1) in the central computer, receiving from one of the plurality of client computers a request to establish a connection;

(2) authenticating, with reference to one of the plurality of authentication tables, that the request received in step (1) is from an authorized client;

(3) responsive to a determination that the request is from an authorized client, allocating resources to establish a virtual private link between the client and a second computer; and

(4) communicating between the authorized client and the second computer using the virtual private link.

18. A method of transparently creating a virtual private network (VPN) between a client computer and a target computer, comprising the steps of:

(1) generating from the client computer a Domain Name Service (DNS) request that requests an IP address corresponding to a domain name associated with the target computer;

> (2) determining whether the DNS request transmitted in step (1) is requesting access to a secure web site; and
>
> (3) in response to determining that the DNS request in step (2) is requesting access to a secure target web site, automatically initiating the VPN between the client computer and the target computer, wherein:
>
> steps (2) and (3) are performed at a DNS server separate from the client computer, and step (3) comprises the step of, prior to automatically initiating the VPN between the client computer and the target computer, determining whether the client computer is authorized to resolve addresses of non secure target computers and, if not so authorized, returning an error from the DNS request.

(Appeal Br., Claims App. i–iii.)

C. Claim Construction

Patent Owner informs us that the '135 Patent has expired. Appeal Br. 6. Requester does not dispute that the '135 Patent is expired, but rather contends the Board need not construe the claims, which, according to Requester, are unpatentable even under Patent Owner's construction. Resp't Br. 2–4.

Because the '135 Patent is expired, we construe the claims using the standard in *Phillips v. AWH Corp.*, 415 F.3d 1303 (Fed Cir. 2005) (en banc) as held in *In re CSB-Sys. Int'l Inc.*, 832 F.3d 1335, 1342 (Fed Cir. 2016) (holding that the PTO should apply the *Phillips* standard for claim construction once a patent expires). We largely agree with Requester, that the claims on appeal do not require construction in order to resolve the issues on appeal as discussed *infra*. However, to the extent claim construction is necessary to resolve the case, we construe the claims under

the *Phillips* standard to the extent necessary as discussed below. *Vivid Techs., Inc. v. Am. Sci. & Eng'g, Inc.*, 200 F.3d 795, 803 (Fed. Cir. 1999).

D. Adopted Rejections

Patent Owner contests the Examiner's decision to reject the claims as follows (Appeal Br. 2–4; *see* RAN 6–10; Ans. 1):

Claim(s) Rejected	35 U.S.C. §	Reference(s)
10, 12–14	102(a)	Aventail Connect v3.1[4]
10, 12–14	102(b)	Aventail Connect v3.01[5]
10, 12, 13	102(b)	AutoSOCKS[6]
11	103(a)	Aventail Connect v3.1, Reed[7]
11	103(a)	Aventail Connect v3.01, Reed
11, 14, 15	103(a)	AutoSOCKS, Reed
16	103(a)	Aventail Connect v3.1, Boden[8]
16	103(a)	Aventail Connect v3.01, Boden
16	103(a)	AutoSOCKS, Reed, Boden
17	103(a)	Aventail Connect v3.1, Weiss[9]
17	103(a)	Aventail Connect v3.01,

[4] *Aventail Connect v3.1/v2.6 Administrator's Guide*, 1999 ("Aventail Connect v3.1").
[5] *Aventail Connect v3.01/v2.51 Administrator's Guide*, 1999 ("Aventail Connect v3.01").
[6] *Aventail AutoSOCKS v2.1 Administration & User's Guide*, 1997 ("AutoSOCKS").
[7] Michael G. Reed, Paul F. Syverson, and David M. Goldschlag, *Proxies for Anonymous Routing*, 12th Annual Computer Security Applications Conference, San Diego, CA, December 9–13, 1996 ("Reed").
[8] Boden et al., US 6,615,357 B1, issued September 2, 2003 ("Boden").
[9] Weiss, US 4,885,778, issued December 5, 1989 ("Weiss").

		Weiss
17	103(a)	AutoSOCKS, Weiss
10, 12, 13, 18	102(a)	Wang[10]
11, 14, 15	103(a)	Wang, Reed
16	103(a)	Wang, Reed, Boden
17	103(a)	Wang, Weiss
10, 12, 13, 18	103(a)	Beser,[11] Kent[12]
18	103(a)	Beser, Kent, Blum[13]
18	103(a)	Beser, Kent, AutoSOCKS
11	103(a)	Beser, Kent, Reed
10, 12–15, 18	102(a)	BinGO[14]
11	103(a)	BinGO, Reed
16	103(a)	BinGO, Borden
17	103(a)	BinGO, Weiss

[10] Wang, Broadband Forum TR-025: Core Network Architecture Recommendations For Access to Legacy Data Networks over ADSL, Issue 1.0 (September 1999), accessible at http://www.broadband-forum.org/technical/download/TR-025.pdf ("Wang").
[11] Beser et al., US 6,496,867 B1, issued December 17, 2002 ("Beser").
[12] S. Kent and R. Atkinson, "Security Architecture for the Internet Protocol," Network Working Group RFC 2401, November 1998 ("Kent").
[13] Blum et al., US 6,182,141 B1, issued January 30, 2001 ("Blum").
[14] BinGO! User's Guide ("BinGO UG") incorporating by reference BinGO! Extended Feature Reference ("BinGO EFR"), accessible at ftp://ftp.funkwerk-ec.com/bintec/old_products/bintec/bingo-generation/bingo/doku/71000b.pdf, ftp://ftp.elmeg.de/bintec/doku/swref/old/71050a_v12.pdf (collectively referred to as "BinGO").

III. PATENT OWNER'S APPEAL

A. Rejections based on Aventail Connect v3.1, Aventail Connect v3.01, and AutoSOCKS ("the Aventail References") (Issues 19–30)[15]

1. The Examiner's Rejections

The Examiner rejected claims 10–17 based on the Aventail References either alone or in combination with other prior art by virtue of adopting the Requester's explanations in the Request. *See* RAN 10–22, citing Request 18–24 and 38–118, Exh. C1–C3. The Examiner did not offer any substantive comments in response to the Appeal Brief or Respondent Brief. Ans. 1.

2. Patent Owner's Contentions

Patent Owner contends, *inter alia*, the Aventail References do not disclose the methods recited in the claims. Appeal Br. 11–25. Patent Owner argues the Federal Circuit has already held that Patent Owner has disclaimed systems such as those described in the Aventail References. *Id.* at 12, citing *VirnetX v. Mangrove Partners*, 778 F. App'x at 909–10.

3. Requester's Contentions

Requester does not dispute Patent Owner's arguments, but rather appears to agree with Patent Owner. *See* Resp't Br. 1 ("Requester recommends that the Board not affirm the Examiner's rejections based on [the Aventail References]."); Reb. Br. 3.

---

[15] The designation "Issue 19" is based on the labeling of rejections by the Patent Owner, Requester, and the Examiner. We refer to these designations for convenience of the reader.

4. Discussion

As a result of the above discussion, it appears Requester has withdrawn its earlier position that claims 10–17 would have been unpatentable over the Aventail References alone or in combination with the other prior art cited. In this regard, we observe that, as discussed above, the Examiner's rejections based on the Aventail References were adopted as set forth in the Request. In addition, the Examiner relies almost exclusively on (and adopts) Requester's previous arguments in maintaining the rejections based on the Aventail References. RAN 28–34.

In view of the above, we reverse the rejections of claims 10–17 over the Aventail References.[16]

B. Anticipation – Wang (Issue 31)

We limit our discussion to claims 10, 13, and 18, which is sufficient to resolve the issues associated with this rejection. *See* 37 C.F.R. § 41.67(c)(1)(vii).

1. Claims 10 and 18

a) The Examiner's Findings

The Examiner adopted the proposed rejection of independent claims 10 and 18 as set forth in the Request. RAN 22, citing Request 12–13,

[16] In this regard, although the Answer indicates all of the grounds in the RAN are maintained based on the determinations made therein (Ans. 1), we cannot reconcile the Examiner's positions, which adopt Requester's rejections and positions, with the positions set forth by the Requester in the Respondent Brief.

25–26, and 119–139, 142; Exh. C4. Thus, the Examiner found Wang discloses client computers (PC Clients), a gateway computer (PC-based remote access server), and a corporate network containing secure destination computers. Request 135, 137–138. The Examiner found Wang discloses systems that include a DNS proxy server, namley L2TP Access Aggregation (LAA) architecture, which includes an L2TP Access Concentrator (LAC) as the DNS proxy server, and point-to-point protocol (PPP) Terminated Aggregation (PTA), which includes a Broadband Access Server (BAS) as the DNS proxy server. Request 135–137; Exh. C4, 24–27. The Examiner found Wang discloses in the LAA architecture, PPP tunnels through the Regional Broadband Network, and the PPP allows for authentication to be requested during negotiation of a configuration request. Request 135–136. The Examiner found Wang discloses the LAC determines the destination of the request based on the user-name and domain information, and also whether a tunnel exists to the proper L2TP Network Server (LNS), and if a tunnel does not exist, the LAC establishes one. Request 136. The Examiner found the BAS provides similar functions in the PTA architecture. Request 136–137.

The Examiner found Wang discloses "VPNs (*i.e.*, secure encrypted tunnels being established after authentication)" that are established in response to DNS requests specifying a secure domain, a corporate network. Request 135–137, citing Wang 14–16, 18; *see id.* at 124, citing Wang 8–9.

b) Patent Owner's Contentions

Patent Owner contends, *inter alia*, that neither the L2TP Access Concentrator (LAC) used in the L2TP Access Aggregation (LAA)

architecture nor the Broadband Access Server (BAS), used in the point-to-point protocol (PPP) Terminated Aggregation (PTA) architecture disclosed in Wang correspond to the DNS proxy server recited in claim 10. Appeal Br. 27. In particular, Patent Owner argues the LAC and BAS do not receive a request from a client computer to look up an IP address for a domain name, but rather, only receive a user name from the user computer during the PPP authentication phase. *Id.* Thus, Patent Owner argues the LAC and BAS challenge the user for a user-name and password, and a user's response to this challenge is not a request to look up an IP address even if the destination Network Service Provider (NSP) is determined by the LAC and BAS after receiving a user-name. *Id.* at 28.

Similarly, Patent Owner argues Wang's LAC and BAS do not disclose the DNS proxy server in claim 10, because neither returns an IP address for a requested domain name if it is determined that access to a non-secure web site has been requested. Appeal Br. 28–29. In this regard, Patent Owner contends the Examiner has already determined Wang does not disclose this feature in reference to claim 3, which recites similar language. *Id.* at 28, citing, e.g., Action Closing Prosecution ("ACP") dated April 27, 2015. Patent Owner contends the LAC determines only the destination Network Service Provider (NSP) from the name, determines if a tunnel already exists to the proper LNS, and if one does not, the LAC establishes a tunnel. *Id.* at 29. Patent Owner contends the BAS does not include additional DNS servers as indicated by Requester, and further that the NSP is not alleged to be the claimed DNS proxy server, such that the NSP cannot correspond to the claimed DNS proxy server. *Id.*

Patent Owner also contends that Wang's LAC and BAS do not generate a request to create a VPN between a client computer and a secure target computer as recited in claim 10. *Id.* at 29–30. Patent Owner argues Requester has not shown that a tunnel exists in any portion of the path between the user computer and the NSP through the BAS such that Wang does not disclose a VPN, and with regard to the LAC, the mere creation of a tunnel does not mean traffic satisfies all the requirements of a VPN. *Id.* at 30. Patent Owner argues that in view of Wang's failure to disclose a VPN, Wang fails to disclose the gatekeeper computer recited in claim 10. *Id.* at 31–32.

Patent Owner contends also that Requester has improperly combined elements from different architectures, the LAC and BAS of Wang, in order to meet the recited DNS proxy server of claim 10. *Id.* at 31.

Regarding independent claim 18, Patent Owner contends Wang does not disclose generating a DNS request from a client computer that requests an IP address corresponding to a domain name associated with a target computer, because Wang merely discloses that a user name is provided by a user to the LAC or BAS (an intermediate device) so that the user's intended NSP is known. *Id.* at 34–36. Patent Owner contends Wang fails to disclose automatically initiating a VPN between a client computer and a target computer in response to determining a DNS request is requesting access to a secure target web site. *Id.* at 36–38. Similar to claim 10, Patent Owner argues that the rejection improperly combines separate elements from Wang to anticipate claim 18. *Id.* at 38.

c) Requester's Contentions

Requester contends the rejection should be affirmed for the reasons stated in the RAN and the Request, as Patent Owner's arguments are substantially the same as those previously of record. Resp't Br. 13.

d) Analysis

We are persuaded by Patent Owner's contentions that Wang does not look up an IP address for a domain name as a result of a request from a client computer as recited in claim 10 and generating from the client computer a DNS request that requests an IP address corresponding to a domain name associated with the target computer as recited in claim 18. Wang discloses that in the LAA embodiment, the user will send a request to initiate a session between a user's Customer Premises Equipment (CPE) and the LAC (DNS proxy server), where for the purposes of authentication, the user enters a user name as well as a domain name for the Network Service Provider (NSP) for example, by entering "Joe@nsp.net." Wang 14–15. The LAC determines the destination NSP "[b]ased on the user-name and domain information." *Id.* at 15. The LAC establishes a tunnel to the proper LNS if one does not exist. *Id.* Thus, as Patent Owner contends, Wang does not disclose the LAC returns an IP address as recited in claim 10. *See* Keromytis Decl. ¶¶ 78, 79.

Similarly, in the PTA architecture, Wang discloses the user initiates a PPP session to the BAS over Asynchronous (ATM) Virtual Circuit Connection (VCC), where the virtual dialer will send a PPP Link Control Protocol (LCP) Configuration-Request to the BAS. Wang 18. The BAS responds with a PPP LCP configuration-ACK and the PPP dialer responds

with a PPP LCP Configuration-ACK to complete the configuration. *Id.* After a link is established, the BAS initiates an authentication stage, which challenges the user for a user-name and password, and the user will reply with a fully qualified domain name. *Id.* Wang discloses "[t]he BAS extracts the domain string portion of the user-name and sends off a query to NSP to authenticate and obtain address information (e.g., DNS server's address)." *Id.* Wang discloses "[i]n the case of IP network, The NSP replies with an IP address and other IP configuration information (e.g. DNS Server's address). This information is passed along to the user during the NCP phase for configuring IP transport (based on IPCP)." *Id.* Thus, as Patent Owner contends, Wang discloses receiving the IP address of the DNS server and not the IP address of a domain name for a target computer. Appeal Br. 35; Keromytis Decl. 80.

Further with respect to claim 10, the Examiner and Requester do not address, and we cannot reconcile, the apparent inconsistency identified by Patent Owner in the Examiner's positions regarding the similar language in claims 3 and 10 as to whether Wang discloses a DNS proxy server that returns an IP address for a requested domain name if it is determined that access to a non-secure web site has been requested. After identifying the proposed rejection of claim 3 from the Request, the Examiner agreed that it was well known for DNS servers to return an IP address associated with a domain name, but found such was not the only mode of operation for the systems disclosed in Wang. ACP 23–24, citing Request 128, Wang 18. The Examiner found the portion of Wang cited in the Request discloses returning an IP address of the DNS server, not the IP address associated with the target domain name. *Id.* at 24. Thus, the Examiner found the Request was

deficient in showing Wang discloses a DNS server that returns an IP address associated with a domain name in response to a query for access to a non secure target and showing that such an operation in Wang would have been inherent. *Id.* Requester did not dispute the Examiner's findings with respect to claim 3. *See* Req. 3rd Comments 18–25 (addressing anticipation rejections of the claims based on Wang, but not the Examiner's findings that the Request failed to show claim 3 was anticipated). The Examiner made no additional comments or findings with respect to claim 10 being anticipated by Wang, choosing instead to rely solely on the Request and Requester's comments with respect thereto. *See* ACP 23, 38; RAN 22, 34, citing Request 12, 13, 25, 26, 119–142, Exh. C4; Req. 3rd Comments 18–25. We observe that the Request, in describing how claims 3 and 10 are anticipated by Wang, relies on some of the same passages therein. *Compare* Request Exh. C4, 11–12, *with* 23–26.

Moreover, we are unable to find a clear statement of how Wang discloses a DNS proxy server that returns an IP address for a requested domain name if it is determined that access to a non-secure web site has been requested as recited in claim 10. *See, e.g.*, Request 12, 13, 25, 26, 119–142, Exh. C4; Req. Req. 3rd Comments 18–25. That is, Requester focuses on whether Wang discloses a DNS proxy server that returns an IP address as a general matter, rather than the specific language of claim 10, as to what happens in the event it is determined the DNS request is for a non-secure web-site.

Therefore, we agree with Patent Owner that the Examiner and Requester have failed to provide sufficient explanation that Wang discloses the system and method recited in claims 10 and 18 as arranged in the claims.

Appeal Br. 31 (citing *NetMoneyIN, Inc. v. VeriSign, Inc.*, 545 F.3d 1359, 1371 (Fed. Cir. 2008)), 36 (citing *Richardson v. Suzuki Motor Co.*, 868 F.2d 1226, 1236 (Fed. Cir. 1989)).

Accordingly, we reverse the Examiner's rejection of claims 10 and 18 as anticipated by Wang.

2. Claim 13

a) The Examiner's Findings

The Examiner adopted the proposed rejection of independent claim 13 as set forth in the Request. RAN 22, citing Request 12–13, 25–26, and 139–142; Ex. C4.

b) Patent Owner's Contentions

Patent Owner argues Wang does not disclose a central computer that maintains a plurality of authentication tables each corresponding to one of the client computers, because authentication tables are not inherently present in the LAC and BAS, which are alleged to correspond to the central computer. Appeal Br. 32–33. Patent Owner argues also that the LAC and BAS in Wang control admissions to tunnels based on the number of users in a tunnel, and not responsive to authorization of a client, such that Wang does not disclose allocating resources to establish a VPN based on a determination that a client is authorized. *Id.* at 33–34.

c) Requester's Contentions

Requester contends the rejection should be affirmed for the reasons stated in the RAN and the Request, as Patent Owner's arguments are substantially the same as those previously of record. Resp't Br. 13.

d) Analysis

We are not persuaded by Patent Owner's arguments. That is, Patent Owner's main argument is that the LAC does not compare the user name with any credential stored therein, and similarly, the BAS does not compare the received user name and password to any credentials stored therein, but rather sends a query to the NSP to authenticate, and as such neither the LAC nor the BAS is a central computer "maintains a plurality of authentication tables" as recited in claim 13.

In responding to Patent Owner's arguments, the Examiner, in the RAN, incorporated by reference Requester's Comments filed July 15, 2015 ("Req. 3rd Comments"). RAN 34. The Examiner and Requester rely on the LAC as a central computer in the LAA architecture, and the BAS as the central computer in the PTA architecture. Request, Exh. C4, 32–33; Req. 3rd Comments 24. The Request identifies the authentication phase between the request from the client and the LAC, and the username and password unique to the user as the information contained in authentication tables on the LAC. Request, Exh. C4, 32, citing Wang, 14–15. In addition, the Request specifically identifies Challenge Handshake Authentication Protocol (CHAP) used in conjunction with the BAS as involving a comparison of credentials stored on the server computer with credentials presented by a client computer as involving the use of authentication tables. Request, Exh. C4, 32–33, citing Wang, 18, Fig. 8. We agree with the Examiner and Requester that claim 13 does not recite any particular form or content for the "authentication tables" recited therein. Req. 3rd Comments 24. As such, we do not discern a distinction between the username and password information that would be stored on the LAC for authentication

phase or the BAS for CHAP authentication and the "authentication tables" recited in claim 13.

As to Patent Owner's arguments that the LAC and BAS in Wang do not allocate resources responsive to a determination that the request is from an authorized client, the Request explains that upon authenticating a client, Wang discloses several gateway computers (LAC, BAS, and NSP) that allocate resources to establish a VPN. Request, Exh. C4, 32–35, citing Wang 12, 14–22. Patent Owner's arguments appear to be based on the position that Wang does not disclose client authorization (Appeal Br. 33–34), which we found unpersuasive as previously discussed. In this regard, whether the LAC and BAS also perform admission control based on the number of users in a tunnel (Appeal Br. 34) does not address the rationale in the Request and adopted by the Examiner.

As a result, we affirm the Examiner's rejection of claim 13 as anticipated by Wang.

C. Obviousness – Wang, Reed, Boden, Weiss (Issues 34–36)

1. Claims 11, 14, and 15 – Wang, Reed (Issue 34)

a) Claims 11 and 14

Regarding claim 11, which depends from claim 10, because we reversed the rejection of claim 10, and as Patent Owner points out, Reed does not remedy the deficiencies of Wang (Appeal Br. 39), we reverse the rejection of claim 11 as obvious over Wang and Reed.

Regarding claim 14, Patent Owner does not set forth any additional arguments with respect to claim 14, relying on it dependency from claim 13 as basis for patentability. Appeal Br. 39. Accordingly, we affirm the

Examiner's rejection of claim 14 for similar reasons as discussed above with respect to claim 13.

b) Claim 15

Claim 15 depends from claim 14 and recites "wherein step (4) comprises the step of comparing an Internet protocol (IP) address in a header of each data packet to a table of valid IP addresses maintained in a table in the second computer."

The Examiner relied on the explanation provided by Requester in the Request that Reed discloses comparing IP addresses because Reed discloses "[w]hen a data cell arrives, the onion router looks up the cell's identifier in its tables and finds the corresponding outbound identifier." RAN 23, citing Request 145–146; Request Exh. C4, 36; Reed 7.

Patent Owner contends Reed does not disclose or suggest the features of claim 15. Appeal Br. 39 (citing § VIII(I)(3), 22). In particular, Patent Owner argues Reed discloses an onion router, which looks up a data cell's identifier in its tables when the data cell arrives at the onion router. *Id.* at 22, citing Reed § 5.3.1. Patent Owner contends the cell identifiers are not IP addresses, because they identify an anonymous connection and not the onion router transmitting a cell. *Id.* citing Reed §§ 5.1, 5.2.1, Fig. 3.

The Examiner and Requester do not respond to Patent Owner's arguments. Resp't Br. 14; Req. Req. 3rd Comments 25.

We agree with Patent Owner that there is no indication in Reed that the cell identifier disclosed therein is an IP address. Reed discloses "cells have type data and are labeled with the identifier of the associated anonymous connection." Reed 7, § 5.3.1. The onion router looks up the cell's identifier in tables within the onion router to find the corresponding

outbound identifier. *Id.* The Examiner and Requester do not sufficiently explain how the identifier would be an IP address. Indeed, Requester relies on the Declaration of Michael Fratto executed on July 7, 2011 (Resp't Br. Exh. E2) for support. *See* Request Exh. C4, 36, citing Fratto Decl. Exh. E2 ¶¶ 193–194. Yet, the cited portion of the Fratto Declaration discussing claim 15 does not rely on Reed for disclosing tables of IP addresses. Fratto Decl. ¶¶ 193–194.

As a result, we reverse the Examiner's rejection of claim 15 as obvious over Wang and Reed.

2. Claim 16 – Wang, Reed, Boden (Issue 35)

Claim 16, depends from claim 15 and is rejected over the combination of Wang, Reed, and Boden. *See* RAN 23; Request 147–148; Exh. C4, 37–38. Thus, because, as Patent Owner points out, Boden does not remedy the deficiencies of Reed (Appeal Br. 39), we reverse the Examiner's rejection of claim 16 for similar reasons as discussed above for claim 15.

3. Claim 17 – Wang, Reed, Weiss (Issue 36)

Regarding claim 17, Patent Owner does not set forth any additional arguments with respect to claim 17, relying on its dependency from claim 13 as the basis for patentability. Appeal Br. 39–40. Accordingly, we affirm the Examiner's rejection of claim 17 for similar reasons as discussed above with respect to claim 13.

D. Obviousness – Beser and Kent (Issues 37–40)

We limit our discussion to claims 10, 13, and 18, which is sufficient to resolve the issues associated with these rejections. *See* 37 C.F.R. § 41.67(c)(1)(vii).

1. Claim 10

a) The Examiner's Findings

In rejecting claim 10 as obvious over Beser and Kent, the Examiner adopted the rejections as proposed by Requestor. RAN 24, citing Request 13, 29–30, and 150–173, Exh. C5.

b) Patent Owner's Contentions

Patent Owner contends, *inter alia*, Beser and Kent do not disclose or suggest both a request to look up an IP address for a domain name and a request to create a VPN as recited in claim 10, because Beser's tunneling request (request to initiate a VoIP association) cannot be both requests, as there is no request from the trusted-third-party network device to the second network device to negotiate the IP tunnel. Appeal Br. 47.

c) Requester's Contentions

Requester argues Beser receives a request from a client computer to look up an IP address for a domain name because Beser discloses that the trusted-third-party-network device, which can be a DNS server, contains a table that correlates unique identifiers (domain names) to both the IP address of the terminating device and the IP address of the associated second network device. *Id.* at 10–11. Requester contends Beser and Kent teach both a request to look up an IP address for a domain name and a request to

create a VPN, because Beser discloses that after the trusted-third-party network device receives a tunneling request, it sends a request to the second network device to begin negotiating the IP tunnel (VPN). *Id.* at 11.

d) Analysis

Claim 10 recites "a gatekeeper computer that allocates resources for the VPN between the client computer and the secure web computer in response to the request by the DNS proxy server." As explained in the '135 Patent, the "[g]atekeeper 2603 can be implemented on a separate computer (as shown in FIG. 26) or as a function within the modified DNS server 2602." '135 Patent, col. 38, ll. 53–55. Claim 10 expressly recites a system including a DNS proxy server *and* a gatekeeper computer that allocates resources for the VPN. Thus, in view of the '135 Patent, we interpret claim 10 to be directed to embodiments where the gatekeeper is a separate computer and not a function within the DNS server.

In the rejection, the Examiner and Requestor rely on the trusted third party network device disclosed in Beser as a gateway computer (*see* Request, Exh. C5, 12, 16) and also as the DNS proxy server (*see* Resp't Br. 9 (discussing how the trusted-third-party network device would return an unsecure IP address, recited in claim 10 as a function performed by the DNS proxy server); Request, Exh. C5, 14). Beser discloses a trusted third party network device 30 that is connected to a public network 12, which may be a domain name server. Beser, col. 3, l. 60 – col. 4, l. 18; Fig. 1. Although Beser discloses the trusted-third party network device 30 may be distributed over several locations (*id.*), the rejection does not sufficiently explain how Beser would be modified to meet the requirements set forth in claim 10.

Thus, because claim 10 requires a gatekeeper computer that allocates resources for the VPN that is separate from a DNS proxy server, we agree with Patent Owner that the same third party trusted network cannot relied on as disclosing both limitations. Appeal Br. 47.

As a result, we reverse the Examiner's rejection of claim 10 and claim 12, dependent therefrom, as obvious over Beser and Kent.

In addition, claim 11, which depends from claim 10, is separately rejected as obvious over Beser, Kent, and Reed (Issue 40). Reed does not remedy the above noted deficiencies (*see* Request, Exh. C5, 16–18). Thus, we reverse the rejection of claim 11 as well.

2. Claim 13

Although claim 13 is listed as rejected in the Request (*see* Request 29) and in the Right of Appeal Notice (*see* RAN 24), we agree with Patent Owner (Appeal Br. 48–49) that there is no articulation of a rejection of claim 13. *See* Request, Exh. C5, 18–19 (showing claim charts for claims 12 and 18, but not claim 13). Neither the Requester nor the Examiner provides any clarity with respect to this issue. *See* RAN *generally*; Resp't Br. 4–14. We are of the view the reference to claim 13 in the rejection is a typographical error.

However, to the extent that the Request and RAN may be interpreted to set forth a rejection of claim 13 as obvious over Beser and Kent, we reverse such a rejection as being facially deficient. That is, there is no articulated prima facie case of how the limitations of claim 13 would have been rendered obvious over Beser and Kent on this record.

3. Claim 18

a) The Examiner's Findings

In rejecting claim 18 as obvious over Beser, Kent, Blum, and AutoSOCKS, the Examiner adopted the rejections as proposed by Requestor. RAN 24–25, 36–37, citing Request 12, 13, 29–32, 150–173, and 178–182, Exh. C5.

b) Patent Owner's Contentions

Patent Owner argues one of ordinary skill in the art would not have combined Beser and Kent because Beser teaches away from encryption between a client computer and a target computer and the IPsec protocol disclosed in Kent. Appeal Br. 42–44, 49. Patent Owner contends Beser and Kent do not disclose or suggest receiving a request from a client computer to look up an IP address for a domain name or determining whether the DNS request is requesting access to a secure web site. *Id.* at 46, 49–51.

Patent Owner contends also that Beser and Kent do not disclose or suggest returning an IP address for a requested domain name if it is determined that access to a non-secure web site has been requested. *Id.* at 44–45. Specifically, Patent Owner contends that Beser is silent as to what would happen if an identifier for an unknown destination were sent to the trusted-third-party network device, and if the destination is unknown, the trusted-third-party network device has no information about the destination and therefore cannot determine whether the tunneling request is requesting access to a secure or unsecure destination or web site. *Id.*

Patent Owner argues Beser and Kent do not disclose or suggest determining whether the client computer is authorized to resolve addresses

of non-secure target computers. *Id.* at 51. Patent Owner contends also that Blum's DNS services do not factor in client authorization. *Id.* at 52–53.

c) Requester's Contentions

Requester points out that the Board, in a number of other proceedings involving other patents in the same family as the '135 Patent, has made findings with respect to Beser that are consistent with the Examiner's determinations in the current reexamination. Resp't Br. 4–5. Requester argues Beser does not teach away from encryption in IP tunneling applications, rather, Beser discloses encryption is conventional and is ordinarily used in IP tunneling schemes, expressly referring to Kent for such implementations. *Id.* at 6–8. Requester argues the practical concerns in implementing encryption in IP tunneling schemes discussed in Beser are in only two data transfer scenarios that may be overcome by using more powerful equipment. *Id.* at 7–8. Requester contends that contrary to Patent Owner's argument that Beser does not disclose returning an IP address if it is determined that access to a non-secure web site is requested, Beser discloses the trusted-third-party network device would return either an unsecure IP address, or if the identifier was not contained in a public DNS entry, a "host unknown" error. *Id.* at 9. Requester argues Beser receives a request from a client computer to look up an IP address for a domain name because Beser discloses that the trusted-third-party-network device, which can be a DNS server, contains a table that correlates unique identifiers (domain names) to both the IP address of the terminating device and the IP address of the associated second network device. *Id.* at 10–11. Requester contends Beser and Kent teach both a request to look up an IP address for a

domain name and a request to create a VPN, because Beser discloses that after the trusted-third-party network device receives a tunneling request, it sends a request to the second network device to begin negotiating the IP tunnel (VPN). *Id.* at 11.

Requester contends Blum discloses errors are provided in a variety of scenarios incidental to DNS queries in its scheme, such as if a DNS request fails or to enforce security policies. Resp't Br. 12. Requester contends that Patent Owner admits various errors are returned in VPN deployment using IPSec as disclosed in Kent including "ICMP error messages." *Id.*

d) Analysis

(1) Collateral Estoppel

At the outset, we observe, as does Requester, that several issues identified by Patent Owner with respect to the combination of Beser and Kent have been decided previously by the Board in other proceedings and subsequently affirmed by the Federal Circuit. In particular, Patent Owner's arguments that Beser teaches away from the use of encryption and the IPsec protocol disclosed in Kent, as well as Patent Owner's arguments that Beser does not render obvious looking up an IP address for a domain name and determining whether the request is requesting access to a secure/non-secure web site have been addressed and as such, we are of the view that Patent Owner is collaterally estopped from relying on such arguments here for the reasons that follow.

As recently held by our reviewing court in *SynQor, Inc. v. Vicor Corp.*, No. 19-1704, slip op. at 13–14, 18 (Fed. Cir. Feb. 22, 2021), collateral estoppel arising from a first reexamination may be applied to a

second reexamination. In addition, as noted in *SynQor*, issue preclusion applies to *inter partes* review. *Id.* at 7, citing *Papst Licensing GmbH & Co. KG v. Samsung Elecs. Am., Inc.*, 924 F.3d 1243, 1250–51 (Fed. Cir. 2019). Thus, as further discussed below, we do not see a reason why issue preclusion would not apply to a reexamination based on a previous *inter partes* review.

> Issue preclusion is appropriate only if: (1) the issue is identical to one decided in the first action; (2) the issue was actually litigated in the first action; (3) resolution of the issue was essential to a final judgment in the first action; and (4) plaintiff had a full and fair opportunity to litigate the issue in the first action.

SynQor, slip op. at 18, quoting *In re Freeman*, 30 F.3d 1459, 1465 (Fed. Cir. 1994).

In determining whether the issues are identical, we observe that the patent claims of the patents at issue in each proceeding need not be identical. *Id.*, quoting *Ohio Willow Wood Co. v. Alps S., LLC*, 735 F.3d 1333, 1342 (Fed. Cir. 2013). In addition, "an 'issue' must be understood broadly enough 'to prevent repetitions litigation of what is *essentially* the same dispute.'" *Id.* at 18–19, citing *B&B Hardware, Inc. v. Hargis Indus., Inc.*, 575 U.S. 138, 157 (2015) (citing Restatement (Second) of Judgments § 27 cmt. *c*, at 252–253).

Here, Patent Owner has argued in multiple *inter partes* review proceedings in which the combination of Beser and Kent have been applied to claims reciting a request for an IP address based on a domain name and where a virtual private network link is established, that Beser does not disclose or suggest DNS look up functionality and that Beser teaches away

from the use of IPsec protocol in Kent, arguments that were rejected by the Board and where the Decisions were affirmed by the Federal Circuit. *See Apple, Inc. v. VirnetX, Inc.*, IPR2016-00331, Paper 29 at 6, 21–41 (PTAB June 22, 2017) (Final Written Decision), *aff'd*, *VirnetX, Inc. v. Apple, Inc.*, 909 F.3d 1375 (Fed. Cir. 2018); *Apple, Inc. v. VirnetX, Inc.*, IPR2015-00810, Paper 44 at 5, 26–45 (PTAB August 30, 2016) (Final Written Decision); *Apple, Inc. v. VirnetX, Inc.*, IPR2015-00812, Paper 43 at 4–5, 16–38 (PTAB August 30, 2016) (Final Written Decision); *Apple, Inc. v. VirnetX, Inc.*, IPR2015-00866, Paper 39 at 4–5, 17–39 (PTAB September 28, 2016) (Final Written Decision); *Apple, Inc. v. VirnetX, Inc.*, IPR2015-00868, Paper 39 at 5, 18–41 (PTAB September 28, 2016) (Final Written Decision); *Apple, Inc. v. VirnetX, Inc.*, IPR2015-00870, Paper 39 at 4–5, 29–52 (PTAB September 28, 2016) (Final Written Decision), *aff'd*, *VirnetX, Inc. v. Apple, Inc.*, 715 F. App'x 1024 (Fed. Cir. 2018). *See VirnetX*, 909 F.3d at 1378 (discussing that a Rule 36 judgment may serve as a basis for collateral estoppel, citing *Phil-Insul Corp. v. Airlite Plastics Co.*, 854 F.3d 1344, 1356–57 (Fed. Cir. 2017)).

Thus, the issues of whether as a general matter, Beser teaches away from encryption and the IPsec protocol disclosed in Kent, whether Beser discloses or renders obvious receiving a request to look up an IP address for a domain name, and whether Beser determines that a requested web-site is secure/non-secure are identical and actually litigated in the aforementioned *inter partes* review proceedings. Such issues were also essential to the final judgment, because they were essential to the determination of whether Beser and Kent rendered the claims obvious. Moreover, Patent Owner had a full and fair opportunity to litigate such issues in the *inter partes* review

proceedings through the presenting of evidence in the form of expert declarations and cross-examination. *See SynQor*, slip op. at 15–17 (citing *MaxLinear, Inc. v. CF CRESPE LLC*, 880 F.3d 1373, 1376 (Fed. Cir. 2018), for the proposition that live testimony is not a prerequisite for the application of collateral estoppel in administrative proceedings such as *inter partes* reviews). Accordingly, Patent Owner is collaterally estopped from arguing that Beser teaches away from encryption and that Beser fails to disclose or render obvious receiving a request to look up an IP address for a domain name and determining whether the request is to a secure/non-secure web site.

Alternatively, even if collateral estoppel does not apply, the findings and rationale of the Examiner and arguments and evidence cited by Requestor, as summarized above, show persuasively that the combination of Beser and Kent renders obvious the claims at issue here. Accordingly, no good reason exists on this record to depart from the above-listed Board decisions directed to materially similar issues.

(2) "determining whether the client computer is authorized . . ."

Claim 18 recites in part, "step (3) comprises the step of, prior to automatically initiating the VPN between the client computer and the target computer, determining whether the client computer is authorized to resolve addresses of non secure target computers and, if not so authorized, returning an error from the DNS request."

The '135 Patent discloses a scenario where if a client does not have permission to establish a normal/non-VPN link, the gatekeeper would reject the request and the DNS proxy server would return an error message to the

client. '135 Patent, col. 40, ll. 4–13. The language of claim 18 is confusing because the language appears to be directed to the disclosed scenario at column 40, lines 4–13 of the '135 Patent, but the claim language does not clearly recite this scenario. In an effort to resolve the issues with respect to claim 18, we consider Patent Owner's arguments and decide whether as a general matter, the prior art would have rendered obvious determining whether a client has permission to access a web site and if not, returning an error message.

We observe, as does Patent Owner, that although claim 18 is listed as rejected as obvious over Beser and Kent alone (RAN 24, Issue 37), and additionally over the combination of Beser, Kent, and Blum (RAN 24, Issue 38), the rejection of claim 18 as set forth in the Request and adopted by the Examiner does not explain how claim 18 would have been obvious over Beser and Kent alone, but rather relies on Blum for the client authorization recitation. *See* Request 176–178, Exh. C5, 19–21. Moreover, regarding the combination of Beser, Kent, and AutoSOCKS (RAN 25, Issue 39; Request 181–182), although the Examiner indicates that the proposed rejection is adopted, as Patent Owner observes, the Examiner expressly found AutoSOCKS does not disclose determining whether the client is authorized to resolve addresses of non-secure target computers. RAN 18.

Accordingly, we limit our discussion to the combination of Beser, Kent, and Blum, and to the extent claim 18 stands rejected over the combination of Beser and Kent alone or the combination of Beser, Kent, and AutoSOCKS, we reverse such rejections due to the lack of a prima facie case.

We are not persuaded by Patent Owner's arguments. In particular, Patent Owner's arguments that Beser does not necessarily require any determination of client computer authorization and that Kent's disclosure of ICMP error messages do not relate to authorization or DNS requests (Appeal Br. 52, citing Keromytis Decl. ¶ 102 and Supp. Keromytis Decl. ¶ 21) fail to appreciate what the prior art as a whole would convey to one of ordinary skill in the art. In this regard, Beser discloses the IP packets may require authentication (Beser, col. 11, ll. 22–25), and Kent discloses access control, which is "prevention of use of a resource in an unauthorized manner." Kent 4, 45. Further, Blum discloses employing protocol filters in order to impart specific capabilities and functions, and when services are not available to a client computer, an error message may be returned to the client. Blum, col. 7, ll. 35–38, 56–58; col. 8, l. 65 – col. 9, l. 3.

Thus, the determination of what web sites, whether secure or non-secure, are available to the client would have been an obvious variation of the authentication procedures disclosed in Beser, Kent, and Blum, such that determining whether a client is authorized to access a non secure target computer and returning an error if the client is not authorized to access a non-secure target as recited in claim 18 would have been obvious.

Accordingly, we affirm the Examiner's rejection of claim 18 as obvious over Beser, Kent, and Blum.

E. Anticipation – BinGO (Issue 41)

We limit our discussion to claims 10, 13, 14, and 18, which is sufficient to resolve the issues associated with this rejection. *See* 37 C.F.R. § 41.67(c)(1)(vii).

1. Claim 10

a) The Examiner's Findings

The Examiner adopted the rejection of claims 10, 12–15, and 18 as anticipated by BinGO as proposed in the Request. *See* RAN 25–26, citing Request 13, 14, 32, 33, and 184–218; Exh. C6. Thus, the Examiner found that BinGO discloses systems including client computers, the BinGO! router or routers in a corporate network acting as a gateway computer, and destinations that are both secure and non-secure. Request 206, citing BinGO 15, Fig. 1-1. The Examiner found BinGO discloses three different ways that will determine if a DNS request specifies a secure destination: (1) a local DNS server is set up on a Local Area Network (LAN); (2) the client computer will compare the DNS request first to the data in the LMHOSTS (LAN Manager HOSTS) file on the client computer; and (3) the BinGO! router is configured to have a corporate network as a "default route," where the DNS request would be resolved. *Id.* at 206–207, citing BinGO UG 17, 40, 145, 175–176, 265–266; BinGO EFR 82–85. The Examiner found BinGO discloses automatically establishing a VPN if a secure destination is requested and if a non-secure web site is requested an IP address of that DNS request is returned to the client computer. *Id.* The Examiner found BinGO discloses the BinGO! router is a gateway computer, because it is separate from the client computer and allocates resources for the VPN between the client computer and the secure web computer in response to the DNS request. *Id.* at 207.

b) Patent Owner's Contentions

Patent Owner contends BinGO does not disclose determining that access to a secure web site has been requested as recited in claim 10. Appeal Br. 55. That is, Patent Owner argues BinGO's disclosures of querying a local DNS server on a local area network (LAN), querying a LMHOSTS file, and querying a "default route" in the absence of any internet service provider (ISP) do not perform the determining step of claim 10. *Id.* Patent Owner argues querying a local DNS server on a LAN and querying a default route in the absence of any ISP, the BinGO! router blindly forwards a request to a primary domain name server in accordance with its "Static Settings." *Id.* at 55–56. As a result, Patent Owner contends the BinGO! router makes no determination as recited in claim 10. *Id.* at 56. Patent Owner contends BinGO discloses that LMHOSTS is an alternative to setting up a domain name server and operates without asking for a DNS. *Id.* Patent Owner contends that BinGO does not disclose a gatekeeper computer that allocates resources for the VPN. *Id.* Patent Owner contends it was improper for the rejection to rely on the combination of BinGO UG and BinGO EFR in order to support an anticipation rejection. *Id.* at 56–57.

c) Requester's Contentions

Requester contends the rejection should be affirmed for the reasons stated in the RAN and the Request, as Patent Owner's arguments are substantially the same as those previously of record. Resp't Br. 14–15.

d) Analysis

Initially, we are not persuaded by Patent Owner's contentions that the rejection improperly relies on the combination of BinGO UG and BinGO EFR in order to support the anticipation rejection. In support of their argument, Patent Owner cites to *Advanced Display Sys., Inc. v. Kent State Univ.*, 212 F.3d 1272, 1282 (Fed. Cir. 2000). Appeal Br. 57. As explained in *Advanced Display*, material not explicitly contained in a single document may still be considered for anticipation if the material is incorporated into the document. 212 F.3d at 1282. The material must be cited "in a manner that makes clear that the material is effectively part of the host document as if it were explicitly contained therein." *Id.* In addition, "the host document must identify with detailed particularity what specific material it incorporates and clearly indicate where that material is found in the various documents." *Id.* "Whether and to what extent material has been incorporated by reference into a host document is a question of law." *Id.* at 1283.

In this case, BinGO UG discloses "BinTec Documentation," which includes the "Extended Feature Reference." BinGO UG 22. BinGO UG expressly refers to BinGO EFR in several places for detailed information in implementing specific functions. *See* BinGO UG 115, 226, 266, 279. Thus, we are of the view that BinGO UG incorporates by reference BinGO EFR. *See* RAN 37; Req. 3rd Comments 40.

Patent Owner's contention that BinGO UG bears a later date (March 1999) than BinGO EFR (February 1999) is not persuasive, as the earlier date in BinGO EFR is consistent with the proposition that in order for the host document to incorporate another document by reference, the incorporated

document must have an earlier date than the host document. *See Advanced Display*, 212 F.3d at 1282–83 (citing *National Latex Prods. Co. v. Sun Rubber Co.,* 274 F.2d 224, 230 (6th Cir. 1959) (requiring a specific reference to material in an earlier application in order have that material considered part of a later application)).

We are not persuaded by Patent Owner's argument that BinGO fails to disclose the "determining" step in claim 10, because Patent Owner's argument is not commensurate in scope with the claim. That is, Patent Owner's contention is based on the position that the "BinGO! router plays no role in deciding where a DNS request is seeking access." Appeal Br. 55. However, claim 10 does not assign the determining step to a particular component or function therein. That is, claim 10 recites that the DNS proxy server returns an IP address for the requested domain name "if it is determined" access to a non-secure web site has been requested. Similarly, claim 10 recites that the DNS proxy server generates a request to create a VPN between the client computer and secure target computer "if it is determined" access to a secure web site has been requested. Thus, whether the BinGO! router plays a role in deciding where a DNS request is seeking access is not relevant to claim 10.

We are also not persuaded by Patent Owner's argument that LMHOSTS does not perform DNS resolution. Appeal Br. 56. BinGO UG expressly discloses: "In the LMHOSTS file, IP addresses are arranged with their computer names in tabular form. If, for example, you are looking for *BossPC*, . . . your PC asks its LMHOSTS file for the corresponding IP address and in this way is able to find the PC." BinGO UG 61. Patent Owner has not sufficiently explained why this name resolution process does

not equate to the DNS resolution process at the level of particularity recited in the claim. *See* Claim 10 ("[T]he DNS proxy server returns the IP address for the requested domain name.").

To the extent Patent Owner's argument is that the processes in BinGO would not distinguish between a secure and non-secure target computer, we are not persuaded, because BinGO expressly discloses connections for "usual internet services" as well as secure connections to a head office. *See* BinGO UG 18.

In addition, BinGO also discloses that in situations where the request from a client computer is a secure target computer, the BinGO! router creates a VPN between the client computer and the target computer. That is, BinGO discloses the BinGO! router sets up a VPN using PPTP (Point to Point Tunneling Protocol) in order to provide access for field service staff via an internet and laptop to a company network. BinGO 226.

As a result, we affirm the Examiner's rejection of claim 10 as anticipated by BinGO.

2. Claim 12

Claim 12 depends from claim 10, and further recites that "the gatekeeper computer determines whether the client computer has sufficient security privileges to create the VPN and, if the client computer lacks sufficient security privileges, rejecting the request to create the VPN."

The Examiner found the BinGO! router discloses requiring authentication before establishing connections to a remote server, by disclosing checking incoming data to decide whether the connection should be allowed, such that acceptance of the call takes place only after correct authentication. Request Exh. C6, citing BinGO UG 269. The Examiner

found BinGO discloses personal authentication and thus discloses the BinGO! router (as the gatekeeper computer) will determine if the client computer has sufficient security privileges to create the VPN and if not, will reject the request to create the VPN. *Id.* citing BinGO UG 77, 114, 145, 149, 154–156, 182, 190–192, 242–243, and 263; BinGO EFR at 76–77, 84–85, 88–91, 93, 95 and 97.

Patent Owner contends the BinGO! router, when checking incoming data, is authenticating the target computer, and does not disclose authenticating its own client computer. Appeal Br. 57–58.

The Examiner incorporates by reference Requester's comments regarding BinGO. RAN 37, citing Req. 3rd Comments 37–45. Requester relies on the rejections as set forth in the Request and contends Patent Owner's arguments in the Appeal Brief are substantially the same as those responded to in the Comments relied upon by the Examiner. *See* Resp't Br. 14–15.

We are not persuaded by Patent Owner's arguments, because as Requester points out, in BinGO, when a user makes a request to the BinGO! router and that request is for a secure connection through a VPN, authentication of the initiating partner is performed via PAP, CHAP, or MS-CHAP. BinGO EFR 84. As such, if the authentication fails, the BinGO! router would reject the request consistent with the very purpose of performing an authentication.

3. Claim 13

a) The Examiner's Findings

Although the Request relies on BinGO's disclosure of local name and common password information, which is also known by a WAN partner and stored on the BinGO! router (the central computer), as the authentication tables (*see* Request, Exh. C6, 30), the Examiner found that it is the use of PAP, CHAP, and MS-CHAP protocols that correspond to the authentication tables recited in claim 13. Req. 3rd Comments 43–44, citing BinGO EFR 84. The Examiner found that such authentication protocols require the BinGO! router as the authenticating device to store user/password combinations in an authentication table, and to use tables to authenticate requests from clients. *Id.*

b) Patent Owner's Contentions

Regarding claim 13, Patent Owner argues that BinGO does not disclose a central computer that maintains a plurality of authentication tables as recited therein. Appeal Br. 58–59. Patent Owner contends there is insufficient explanation as to how the use of PAP, CHAP, and MS-CHAP protocols in BinGO necessarily require a central computer that maintains a plurality of authentication tables. *Id.* Patent Owner contends that BinGO's disclosure of checking incoming calls does not constitute determining that the request is from an authorized client or allocating resources to establish a VPN, because BinGO does not disclose a BinGO! router that authenticates its own client. *Id.* at 59. In addition, Patent Owner contends the rejection relies on BinGO EFR for this feature, which is an additional reference that cannot be combined with BinGO UG in an anticipation rejection. *Id.*; *see id.*

at 56–57. Patent Owner argues that BinGO does not disclose any virtual private link or encrypted communications. *Id.*

c) Requester's Contentions

The Examiner incorporates by reference Requester's comments regarding BinGO. RAN 37, citing Req. 3rd Comments 37–45. Requester relies on the rejections as set forth in the Request and contends Patent Owner's arguments in the Appeal Brief are substantially the same as those responded to in the Comments relied upon by the Examiner. *See* Resp't Br. 14–15.

d) Analysis

Initially, we have already addressed the rejection's reliance on the combination of BinGO UG and BinGO EFR above with respect to claim 10. We are not persuaded by Patent Owner's argument that BinGO does not disclose a central computer with authentication tables. Claim 13 recites the central computer receives a request from a client computer to establish a connection, and requires a determination that the request is from an authorized client.

In BinGO, when a user makes a request to the BinGO! router and that request is for a secure connection through a VPN, authentication of the initiating partner is performed via PAP, CHAP, or MS-CHAP. BinGO EFR 84. As discussed above with respect to Wang, we do not discern a distinction between the "authentication tables" recited in claim 13 and such a disclosure as in BinGO. Once this authentication has taken place, the BinGO! router can set up a VPN (allocate resources) between the client and second computer. BinGO UG 226.

Accordingly, we affirm the Examiner's rejection of claim 13 as anticipated by BinGO.

4. Claims 14 and 15

Claim 14 depends from claim 13 and recites "wherein step (4) comprises the step of communicating according to a scheme by which at least one field in a series of data packets is periodically changed according to a known sequence."

In adopting the rejections set forth in the Request, the Examiner found the BinGO! router "can be configured to implement a variety of Network Address Translation (NAT) schemes." Request 212, Exh. C6, 33. The Examiner found "NAT schemes inherently function by changing at least one field in a series of data packets periodically according to a known sequence." *Id.* The Examiner relied on an additional reference, RFC 2663, as "representative of what was publicly known about NAT schemes." *Id.*; *see* Resp't Br. Exh. Y17. The Examiner found RFC 2663 discloses NAT devices provide a transparent routing solution to end hosts by modifying end node addresses en-route. *Id.* citing RFC 2663 1. The Examiner found NAT schemes were known to be useful and compatible with VPNs based on IP tunneling. *Id.* citing RFC 2663 21.

Patent Owner contends BinGO does not disclose NAT as being used in a virtual private link. Appeal Br. 59. Patent Owner contends further that BinGO does not incorporate by reference RFC 2663, or explain that its use of NAT complies with RFC 2663. *Id.* citing BinGO UG 244–49.

Requester contends NAT schemes inherently function by periodically changing at least one field in a series of data packets according to a predetermined sequence. Req. 3rd Comments 45, citing Request 212; RAN

37. Requester contends "[b]ecause the BinGO! router is able to map IP addresses so that the datagrams are sent to the correct end-point (as opposed to a random end-point), the sequence of address changes must be known." *Id.* Requester contends that BinGO explicitly discloses it uses NAT, which is a standard defined in RFC 2663, such that BinGO incorporates RFC 2663 by reference. *Id.*

We are persuaded by Patent Owner's argument that BinGO does not inherently disclose the use of NAT in a manner that necessarily complies with RFC 2663 by incorporating RFC 2663 by reference. That is, BinGO has a copyright date of March 1999. BinGO UG 3. RFC 2663 has a date of August 1999. RFC 2663 1. Thus, BinGO was available prior to RFC 2663, and cannot have incorporated RFC 2663 by reference. In addition, BinGO expressly discloses the guidelines and standards to which it adheres, and RFC 2663 is not listed. BinGO UG 2–3.

To the extent the rejection of claim 14 relies on BinGO itself, we find no express description that in using NAT, the requirement in claim 14 that "at least one field in a series of data packets is periodically changed according to a known sequence" is disclosed therein. BinGO UG 244–249.

Thus, the Examiner's and Requester's position that claim 14 is anticipated by BinGO via the incorporation of RFC 2663 is not sufficiently supported. Accordingly, we reverse the rejection of claim 14 as anticipated by BinGO. Because claim 15 depends from claim 14, we reverse the rejection of claim 15 as well.

5. Claim 18

a) Patent Owner's Contentions

Patent Owner contends BinGO does not disclose determining that access to a secure web site has been requested as recited in claim 18. Appeal Br. 55, 60. That is, Patent Owner argues BinGO's disclosures of querying a local DNS server on a local area network (LAN), querying a LMHOSTS file, and querying a "default route" in the absence of any internet service provider (ISP) do not perform the determining step of claim 18. *Id.* Patent Owner argues that in querying a local DNS server on a LAN and querying a default route in the absence of any ISP, the BinGO! router blindly forwards a request to a primary domain name server in accordance with its "Static Settings." *Id.* at 55–56. As a result, Patent Owner contends the BinGO! router makes no determination as recited in claim 18. *Id.* at 56. Patent Owner contends BinGO discloses that the LMHOSTS is an alternative to setting up a domain name server and operates without asking for a DNS. *Id.* Patent Owner contends BinGO does not disclose a request to create a VPN between the client computer and a secure target computer as BinGO does not mention encryption or VPNs at all. *Id.* Patent Owner contends that the rejection improperly relies on a second reference, BinGO EFR, which is improper for an anticipation rejection. *Id.* at 56–57. Patent Owner contends that the VPN disclosed in BinGO EFR is implemented only for connecting to the Internet through an ISP, which is the result of a query requesting a non-secure computer, not for a secure web site as recited in claim 18. *Id.* at 57.

In addition, Patent Owner argues BinGO does not disclose that steps (2) and (3) are performed at a DNS server separate from a client computer as

recited in claim 18. Appeal Br. 60. In particular, Patent Owner contends the BinGO! router is not a DNS server, and that a DNS server located on either a LAN or corporate network does not automatically initiate a VPN. *Id.* at 60–61. Patent Owner argues also that BinGO does not disclose determining whether a client computer is authorized to resolve addresses of non secure target computers, and if not so authorized, returning an error from the DNS request, and the position that such a step would be inherent in the BinGO! router is not sufficiently supported. *Id.* at 61–62.

The Examiner and Requester did not provide any specific responses to Patent Owner's contentions.

b) Discussion

BinGO discloses that its BinGO! router is used as a DNS proxy server. BinGO 87. BinGO discloses the BinGO! router is used to connect to non-secure sites over the internet, as well as secure sites such as a company's head office. *See* BinGO 15, Fig. 1-1, 18. BinGO discloses that when the user enters a DNS request for a website (www.bintec.de), if the destination is unknown to the BinGO! router, it forwards the request to a further DNS server (provider) for domain name resolution. *Id.* at 92. BinGO discloses the IP address is returned to the PC and the PC is connected to the destination using a default route. *Id.*

Although in this case, the BinGO! router itself does not perform the domain name resolution step, claim 18 allows for such a scenario. That is, claim 18 recites only that that "steps (2) and (3) are performed at a DNS server separate from the client computer." Claim 18 does not require a particular DNS server to perform determining step (2), which allows for the

BinGO! router acting as a DNS proxy server to forward the request to another DNS as disclosed in BinGO.

BinGO also discloses that in situations where the request from a client computer is a secure target computer, the BinGO! router creates a VPN between the client computer and the target computer. That is, BinGO disclose the BinGO! router sets up a VPN using PPTP (Point to Point Tunneling Protocol) in order to provide access for field service staff via an internet and laptop to a company network. BinGO 226. This also does not violate the language of claim 18, which again only recites that step (3) is performed at a DNS server separate from the client computer.

As to determining whether the client computer is authorized to access non-secure target computers and returning an error message if it determined the client is not authorized to access non-secure target computers, we reiterate the issues with claim 18 discussed above. As we discussed above, BinGO discloses authentication of the initiating partner, which would include situations where the requested web-site is non-secure. Although it is true that BinGO does not expressly disclose returning an error message should the authentication fail, we do not find it credible to argue that one of ordinary skill in the art would not have immediately understood that such an authentication failure would result in an error message being returned as contended by Requester. Req. 3rd Comments 41.

F. Obviousness – Claim 11 – BinGO, Reed (Issue 42)

Claim 11 depends from claim 10, and further recites "wherein the gatekeeper computer creates the VPN by establishing an IP address hopping

regime that is used to pseudorandomly change IP addresses in packets transmitted between the client computer and the secure target computer."

The Examiner adopted the rejection of claim 11 as proposed by the Requester. Request 33, Exh. C6, 26; RAN 26. The Examiner found BinGO does not expressly disclose an IP hopping scheme, but Reed discloses an IP hopping scheme in the form of an onion-routing scheme that meets the requirements of the IP address hopping regime recited in claim 11. Request, Exh. C6, 26–27. The Examiner determined one of ordinary skill in the art would have been motivated by BinGO to ensure that pathways carrying sensitive data are shielded from the interception or monitoring and would have integrated the onion-routing schemes disclosed in Reed into a VPN solution based on use of BinGO! routers. *Id.* at 27.

Patent Owner contends Reed is incompatible with direct-dialing in BinGO, and if Reed is combined with the non-direct dialing in BinGO, there is no transparent creation of a VPN as recited in claim 10. Appeal Br. 62.

We are not persuaded by Patent Owner's contentions. As the Examiner stated, despite Patent Owner's arguments pointing out differences between the two references, Patent Owner does not provide a sufficient explanation as to why the results of the combination would be unsatisfactory or that the principle of operation would be changed in a manner that would render the combination inoperable. RAN 38. Although Patent Owner suggests that the Examiner and Requester now rely on indirect dialing, we do not find the Examiner's and Requester's position to be so limited. Requester and the Examiner stated only that BinGO does not suggest that a direct dial is the sole intended purpose of its communication schemes, that any other method for transporting packets would be unsatisfactory. *Id.*; Req.

3rd Comments 45. Patent Owner has not provided a sufficient explanation to demonstrate error in the rejection's reasoning that one of ordinary skill in the art would have applied the onion routing scheme of Reed in a VPN solution in BinGO in a manner that would have rendered claim 11 obvious.

We affirm the rejection of claim 11 as obvious over BinGO and Reed.

G. Obviousness – Claim 16 – BinGO, Boden (Issue 43)

Claim 16 depends from claim 15, and is rejected over the combination of BinGO and Boden. Request 33–34; RAN 26. Because Boden does not rectify the deficiencies discussed above with respect to claim 15 (dependent from claim 14), we reverse the rejection of claim 16 as obvious over BinGO and Boden.

H. Obviousness – Claim 17 – BinGO, Weiss

Claim 17 depends from claim 13 and further recites "wherein step (2) comprises the step of using a checkpoint data structure that maintains synchronization of a periodically changing parameter known by the central computer and the client computer to authenticate the client."

1. The Examiner's Findings

The Examiner found BinGO discloses BinGO! routers may be configured to use SecurID token authentication methods. Request, Exh. C6, 36, citing BinGO EFR 56; RAN 26–27. The Examiner found BinGO discloses a configuration where a checkpoint is included, which is used by the BinGO! router to implement a SecurID authentication process. *Id.* citing BinGO EFR 56. The Examiner found Weiss discloses methods in which codes are periodically changed according to a pre-defined algorithm and at

least one dynamic variable. *Id.* at 36–37, citing Weiss, col. 1, ll. 63–68. The Examiner found Weiss discloses the method can be used with a mechanism that synchronizes the codes for use in a method for authenticating users. *Id.* at 37, citing Weiss, col. 2, l. 44 – col. 3, l. 27, col. 3, l. 38 – col. 4, l. 29. The Examiner found Weiss discloses the code is represented in a data structure, and used by client server computers to perform authentication of a user. *Id.* citing Weiss, col. 5, ll. 34–43, col. 6, ll. 9–17, col. 7, ll. 14–32.

The Examiner found the SecurID token system is an example of periodically changing the token system that is synchronized by the client and server as taught by Weiss. *Id.* The Examiner determined that because BinGO EFR provides specific direction to incorporate token authentication processes as disclosed in Weiss into the BinGO! router, the additional step of using a checkpoint data structure that maintains synchronization of a periodically changing parameter known by the central computer and the client computer to authenticate the client would have been obvious. *See id.*

2. Patent Owner's Position

Patent Owner contends that the Examiner's reasoning for combining Weiss and BinGO is not sufficiently supported because the discussion in BinGO EFR of SecurID features is for a BRICK router and not for the BinGO! router. Appeal Br. 64. Patent Owner also argues that BinGO UG and BinGO EFR are not properly combinable for the reasons discussed above. *Id.*

3. Requester's Contentions

The Examiner and Requester maintain that BinGO and Weiss are properly combinable. RAN 39; Req. 3rd Comments 46; Resp't Br. 15.

4. Analysis

As discussed above for claim 13, we are of the view that BinGO UG incorporates BinGO EFR by reference. We are also not persuaded by Patent Owner's contention that the SecurID features disclosed in BinGO EFR are for a BRICK router and not for the BinGO! router. That is, BinGO EFR discloses expressly that it is applicable to both BIANCA/BRICK and BinGO! routers. BinGO EFR "NOTE" on page following cover page. Thus, we are not persuaded that one of ordinary skill in the art would have understood the SecurID feature as not applicable to the BinGO! router as well as the BRICK router.

Accordingly, we affirm the Examiner's rejection of claim 17 as obvious over the combination of BinGO and Weiss.

IV. CONCLUSION

In summary, the status of the Adopted Rejections is as follows:

Claim(s) Rejected	35 U.S.C. §	Reference(s)/Basis	Affirmed	Reversed
10, 12–14	102(a)	Aventail Connect v3.1		10, 12–14
10, 12–14	102(b)	Aventail Connect v3.01		10, 12–14
10, 12, 13	102(b)	AutoSOCKS		10, 12, 13
11	103(a)	Aventail Connect v3.1, Reed		11
11	103(a)	Aventail Connect v3.01, Reed		11
11, 14, 15	103(a)	AutoSOCKS, Reed		11, 14, 15
16	103(a)	Aventail Connect v3.1, Boden		16
16	103(a)	Aventail Connect v3.01, Boden		16
16	103(a)	AutoSOCKS, Reed, Boden		16
17	103(a)	Aventail Connect v3.1, Weiss		17
17	103(a)	Aventail Connect v3.01, Weiss		17
17	103(a)	AutoSOCKS, Weiss		17
10, 12, 13, 18	102(a)	Wang	13	10, 12, 18
11, 14, 15	103(a)	Wang, Reed	14	11, 15
16	103(a)	Wang, Reed, Boden		16
17	103(a)	Wang, Weiss	17	

10, 12, 13, 18	103(a)	Beser, Kent		10, 12, 13, 18
18	103(a)	Beser, Kent, Blum	18	
18	103(a)	Beser, Kent, AutoSOCKS		18
11	103(a)	Beser, Kent, Reed		11
10, 12–15, 18	102(a)	BinGO	10, 12, 13, 18	14, 15
11	103(a)	BinGO, Reed	11	
16	103(a)	BinGO, Borden		16
17	103(a)	BinGO, Weiss	17	
Overall Outcome			**10–14, 17, 18**	**15, 16**

In accordance with 37 C.F.R. § 41.79(a)(1), the "[p]arties to the appeal may file a request for rehearing of the decision within one month of the date of: . . . [t]he original decision of the Board under § 41.77(a)." A request for rehearing must be in compliance with 37 C.F.R. § 41.79(b). Comments in opposition to the request and additional requests for rehearing must be in accordance with 37 C.F.R. § 41.79(c) & (d), respectively. Under 37 C.F.R. § 41.79(e), the times for requesting rehearing under paragraph (a) of this section, for requesting further rehearing under paragraph (d) of this section, and for submitting comments under paragraph (c) of this section may not be extended.

An appeal to the United States Court of Appeals for the Federal Circuit under 35 U.S.C. §§ 141–144 and 315 and 37 C.F.R. § 1.983 for an *inter partes* reexamination proceeding "commenced" on or after November 2, 2002 may not be taken "until all parties' rights to request rehearing have been exhausted, at which time the decision of the Board is final and

appealable by any party to the appeal to the Board." 37 C.F.R. § 41.81. *See also* MPEP § 2682 (8th ed., Rev. 7, July 2008).

AFFIRMED IN PART

Appeal 2021-001315
Reexamination Control 95/001,682
Patent 6,502,135 B1

PATENT OWNER:

PAUL HASTINGS LLP
2050 M. Street NW
Washington DC 20036

THIRD PARTY REQUESTER:

SIDLEY AUSTIN LLP
2021 McKinney Avenue
Suite 2000
Dallas, TX 75201

UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
95/001,682	07/11/2011	6,502,135	077580-0132	1074

137313 7590 01/10/2022

PAUL HASTINGS LLP
2050 M. Street NW
Washington, DC 20036

EXAMINER	
PEIKARI, BEHZAD	

ART UNIT	PAPER NUMBER
3992	

MAIL DATE	DELIVERY MODE
01/10/2022	PAPER

Please find below and/or attached an Office communication concerning this application or proceeding.

The time period for reply, if any, is set in the attached communication.

UNITED STATES PATENT AND TRADEMARK OFFICE

BEFORE THE OFFICE OF THE UNDERSECRETARY AND DIRECTOR OF THE UNITED STATES PATENT AND TRADEMARK OFFICE

APPLE INC.,
Requester

v.

Patent of VIRNETX INC.,
Patent Owner and Appellant

Appeal 2021-001315
Reexamination Control 95/001,682
Patent 6,502,135 B1
Technology Center 3900

Before ANDREW HIRSHFELD, *Commissioner for Patents, Performing the Functions and Duties of the Under Secretary of Commerce for Intellectual Property and Director of the United States Patent and Trademark Office.*

ORDER

Patent Owner, VirnetX Inc. ("VirnetX"), filed a Request for Rehearing ("Request") in the above-captioned case. *See* Reexamination Control No. 95/001,682, Patent Owner Request, dated June 14, 2021. In the Request, "VirnetX respectfully submits that whatever remedy is provided by the Supreme Court [in *United States v. Arthrex*, Nos. 19-1434, 19-1452, 19-1458] should be provided in the present reexamination." *Id.* at 6. After the Supreme Court issued its decision in *Arthrex*, the Patent Trial and Appeal Board ("Board") referred VirnetX's request to Mr. Hirshfeld, Commissioner for Patents, Performing the Functions and Duties of the Under Secretary of Commerce for Intellectual Property and Director of the United States Patent and Trademark Office.

It is ORDERED that the Request is denied; and

FURTHER ORDERED that the Board's Decision on Appeal in this case is the final decision of the agency.

Appeal 2021-001315
Reexamination Control 95/001,682
Patent 6,502,135

For Patent Owner:

PAUL HASTINGS LLP
2050 M STREET NW
WASHINGTON, DC 20036

For Third Party Requester:

HAYNES AND BOONE, LLP
IP SECTION
2323 VICTORY AVENUE, SUITE 700
DALLAS, TX 75219

US006502135B1

(12) **United States Patent**
Munger et al.

(10) **Patent No.: US 6,502,135 B1**
(45) **Date of Patent: Dec. 31, 2002**

(54) **AGILE NETWORK PROTOCOL FOR SECURE COMMUNICATIONS WITH ASSURED SYSTEM AVAILABILITY**

(75) Inventors: **Edmund Colby Munger**, Crownsville, MD (US); **Douglas Charles Schmidt**, Severna Park, MD (US); **Robert Dunham Short, III**, Leesburg, VA (US); **Victor Larson**, Fairfax, VA (US); **Michael Williamson**, South Riding, VA (US)

(73) Assignee: **Science Applications International Corporation**, San Diego, CA (US)

(*) Notice: Subject to any disclaimer, the term of this patent is extended or adjusted under 35 U.S.C. 154(b) by 0 days.

(21) Appl. No.: **09/504,783**

(22) Filed: **Feb. 15, 2000**

Related U.S. Application Data

(63) Continuation-in-part of application No. 09/429,643, filed on Oct. 29, 1999
(60) Provisional application No. 60/106,261, filed on Oct. 30, 1998, and provisional application No. 60/137,704, filed on Jun. 7, 1999.

(51) **Int. Cl.**[7] **G06F 15/173**
(52) **U.S. Cl.** **709/225**; 709/229; 709/245
(58) **Field of Search** 709/249, 223, 709/225, 229, 245; 713/201

(56) **References Cited**

U.S. PATENT DOCUMENTS

4,933,846 A 6/1990 Humphrey et al.

(List continued on next page.)

FOREIGN PATENT DOCUMENTS

DE	0 838 930	12/1999
DE	199 24 575	12/1999
EP	2 317 792	4/1998
EP	0 858 189	8/1998
GB	0 814 589	12/1997
WO	WO 98/27783	6/1998
WO	WO 98 59470	12/1998
WO	WO 99 38081	7/1999
WO	WO 99 48303	9/1999
WO	WO 00/70458	11/2000
WO	WO 01 50688	7/2001

OTHER PUBLICATIONS

Fasbender, Kesdogan, and Kubitz: "Variable and Scalable Security: Protection of Location Information in Mobile IP", IEEE publication, 1996, pp. 963–967.

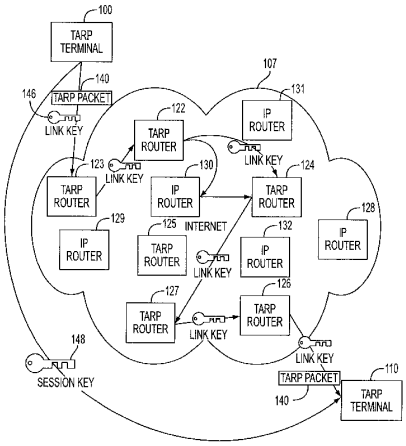
(List continued on next page.)

Primary Examiner—Krisna Lim
(74) *Attorney, Agent, or Firm*—Banner & Witcoff, Ltd.

(57) **ABSTRACT**

A plurality of computer nodes communicate using seemingly random Internet Protocol source and destination addresses. Data packets matching criteria defined by a moving window of valid addresses are accepted for further processing, while those that do not meet the criteria are quickly rejected. Improvements to the basic design include (1) a load balancer that distributes packets across different transmission paths according to transmission path quality; (2) a DNS proxy server that transparently creates a virtual private network in response to a domain name inquiry; (3) a large-to-small link bandwidth management feature that prevents denial-of-service attacks at system chokepoints; (4) a traffic limiter that regulates incoming packets by limiting the rate at which a transmitter can be synchronized with a receiver; and (5) a signaling synchronizer that allows a large number of nodes to communicate with a central node by partitioning the communication function between two separate entities.

17 Claims, 35 Drawing Sheets



U.S. PATENT DOCUMENTS

5,588,060	A		12/1996	Aziz	
5,689,566	A		11/1997	Nguyen	
5,796,942	A		8/1998	Esbensen	
5,805,801	A		9/1998	Holloway et al.	
5,842,040	A		11/1998	Hughes et al.	
5,878,231	A	*	3/1999	Baehr et al.	709/243
5,892,903	A		4/1999	Klaus	
5,898,830	A	*	4/1999	Wesinger et al.	709/225
5,905,859	A		5/1999	Holloway et al.	
6,006,259	A		12/1999	Adelman et al.	
6,016,318	A	*	1/2000	Tomoike	370/338
6,052,788	A		4/2000	Wesinger, Jr. et al.	
6,079,020	A	*	6/2000	Liu	713/201
6,119,171	A		9/2000	Alkhatib	
6,178,505	B1	*	1/2001	Schneider et al.	713/168
6,226,751	B1	*	5/2001	Arrow et al.	370/351
6,243,749	B1		6/2001	Sitaraman et al.	
6,286,047	B1	*	9/2001	Ramanathan et al.	345/733
6,330,562	B1	*	12/2001	Boden et al.	707/10
6,332,158	B1	*	12/2001	Risley et al.	709/219
6,353,614	B1	*	3/2002	Borella et al.	370/389

OTHER PUBLICATIONS

Linux FreeS/WAN Index File, printed from http://liberty-.freeswan.org/freeswan_trees/freeswan–1.3/doc/ on Feb. 21, 2002, 3 pages.

J. Gilmore, "Swan: Securing the Internet against Wiretapping", printed from http://liberty.freeswan.org/freeswan_trees/freesswan–1.3/doc/rationale.html on Feb. 21, 2002, 4 pages.

Glossary for the Linux FreeS/WAN project, printed from http://liberty.freeswan/org/freeswan_trees/freeswan–1.3/doc/glossary.html on Feb. 21, 2002, 25 pages.

Alan O. Frier et al., "The SSL Protocol Version 3.0", Nov. 18, 1996, printed from http://www.netscape.com/eng/ss13/draft302.txt on Feb. 4, 2002, 56 pages.

Reiter, Michael K. and Rubin, Aviel D. (AT&T Labs—Research), "Crowds: Anonymity for Web Transactions", pp. 1–23.

Dolev, Shlomi and Ostrovsky, Rafail, "Efficient Anonymous Multicast and Reception" (Extended Abstract), 16 pages.

Rubin, Aviel D., Geer, Daniel, and Ranum, Marcus J. (Wiley Computer Publishing), "Web Security Sourcebook", pp. 82–94.

Shree Murthy et al., "Congestion–Oriented Shortest Multipath Routing", Proceedings of IEEE INFOCOM, 1996, pp. 1028–1036.

Jim Jones et al., "Distributed Denial of Service Attacks: Defenses", Global Integrity Corporation, 2000, pp. 1–14.

Search Report (dated Jun. 18, 2002), International Application No. PCT/US01/13260.

Search Report (dated Jun. 28, 2002), International Application No. PCT/US01/13261.

Donald E. Eastlake, "Domain Name System Security Extensions", DNS Security Working Group, Apr. 1998, 51 pages.

D. B. Chapman et al., "Building Internet Firewalls", Nov. 1995, pp. 278–297 and pp. 351–375.

P. Srisuresh et al., "DNS extensions to Network Address Translators", Jul. 1998, 27 pages.

Laurie Wells, "Security Icon", Oct. 19, 1998, 1 page.

W. Stallings, "Cryptography And Network Security", 2nd Edition, Chapter 13, IP Security, Jun. 8, 1998, pp. 399–400.

W. Stallings, "New Cryptography and Network Security Book", Jun. 8, 1998, 3 pages.

* cited by examiner

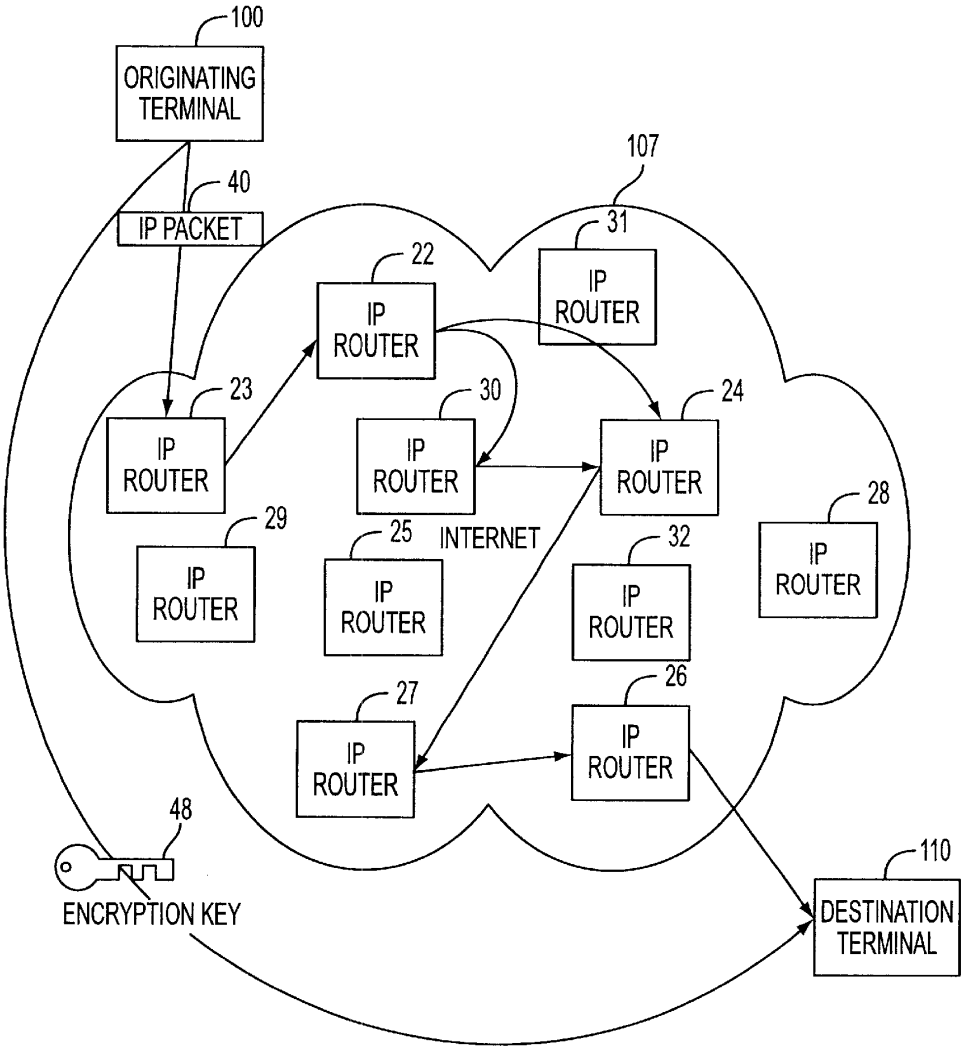

100
ORIGINATING TERMINAL
40
IP PACKET
107
31
IP ROUTER
22
IP ROUTER
23
IP ROUTER
30
IP ROUTER
24
IP ROUTER
28
IP ROUTER
29
IP ROUTER
25
IP ROUTER
INTERNET
32
IP ROUTER
27
IP ROUTER
26
IP ROUTER
48
ENCRYPTION KEY
110
DESTINATION TERMINAL

FIG. 1

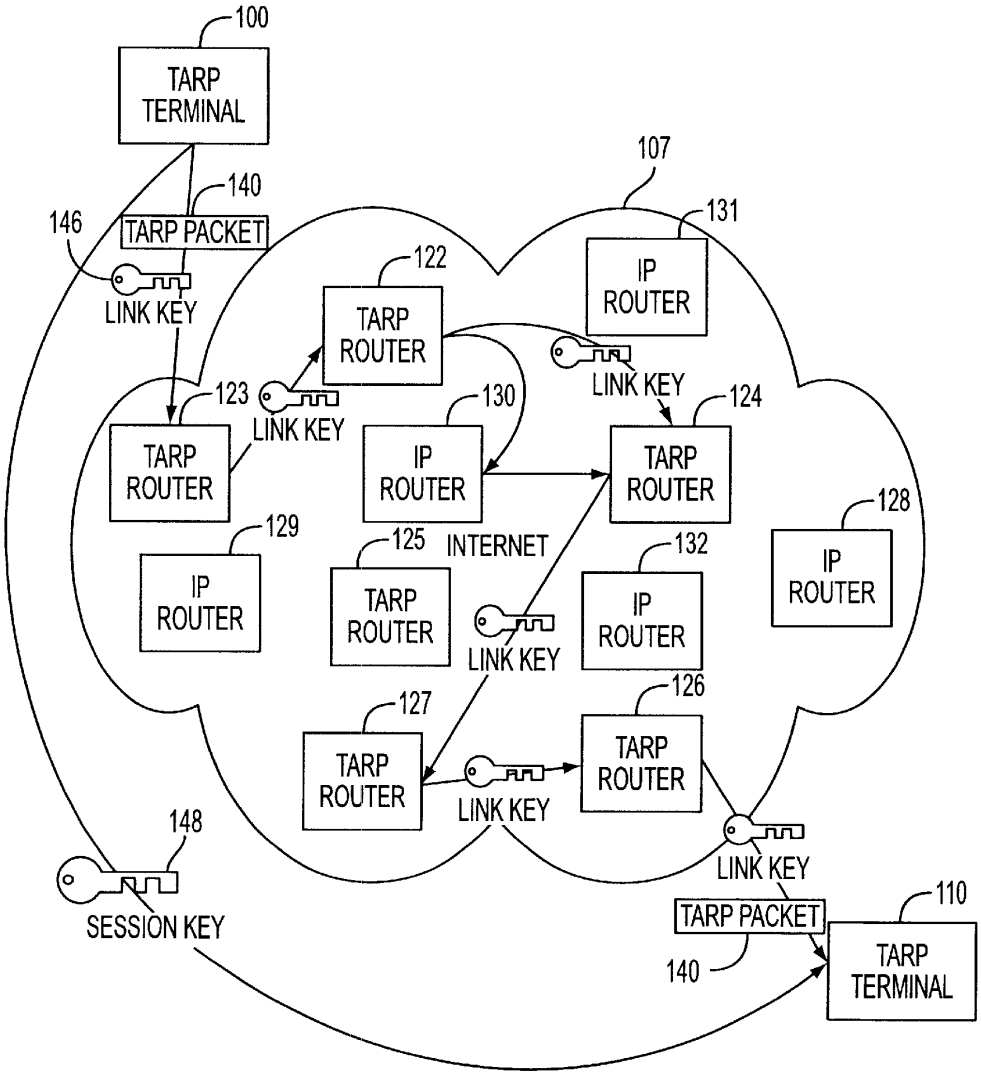

100
TARP TERMINAL
140
TARP PACKET
146
LINK KEY
107
131
IP ROUTER
122
TARP ROUTER
LINK KEY
123
LINK KEY
130
124
TARP ROUTER
IP ROUTER
TARP ROUTER
128
129
IP ROUTER
125
INTERNET
132
TARP ROUTER
IP ROUTER
IP ROUTER
LINK KEY
126
127
TARP ROUTER
TARP ROUTER
LINK KEY
148
LINK KEY
SESSION KEY
110
TARP PACKET
140
TARP TERMINAL

FIG. 2

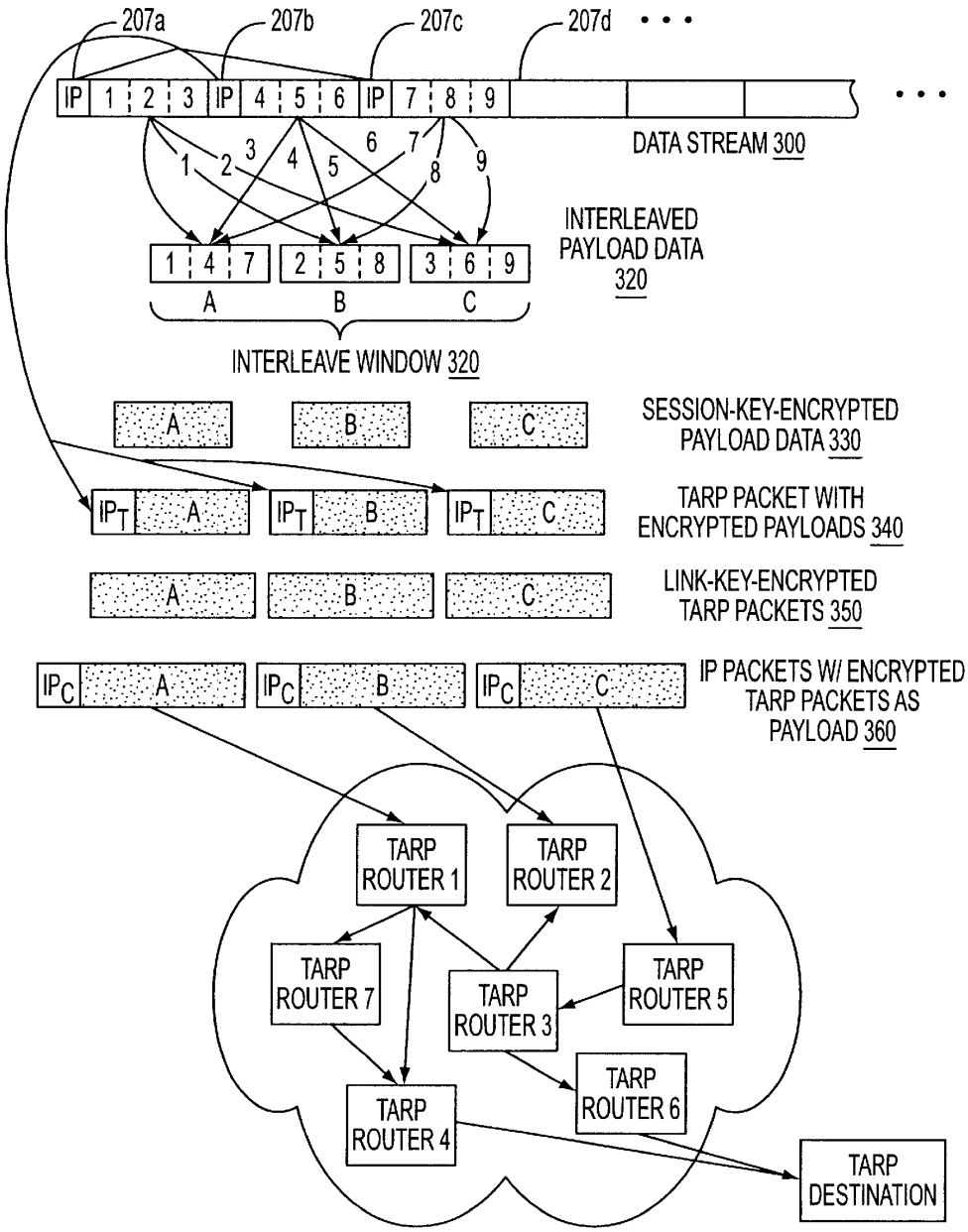

207a
207b
207c
207d
IP 1 2 3 IP 4 5 6 IP 7 8 9
DATA STREAM 300
INTERLEAVED PAYLOAD DATA 320
1 4 7
2 5 8
3 6 9
A
B
C
INTERLEAVE WINDOW 320
SESSION-KEY-ENCRYPTED PAYLOAD DATA 330
IPT A
IPT B
IPT C
TARP PACKET WITH ENCRYPTED PAYLOADS 340
LINK-KEY-ENCRYPTED TARP PACKETS 350
IPC A
IPC B
IPC C
IP PACKETS W/ ENCRYPTED TARP PACKETS AS PAYLOAD 360
TARP ROUTER 1
TARP ROUTER 2
TARP ROUTER 7
TARP ROUTER 3
TARP ROUTER 5
TARP ROUTER 6
TARP ROUTER 4
TARP DESTINATION

FIG. 3A

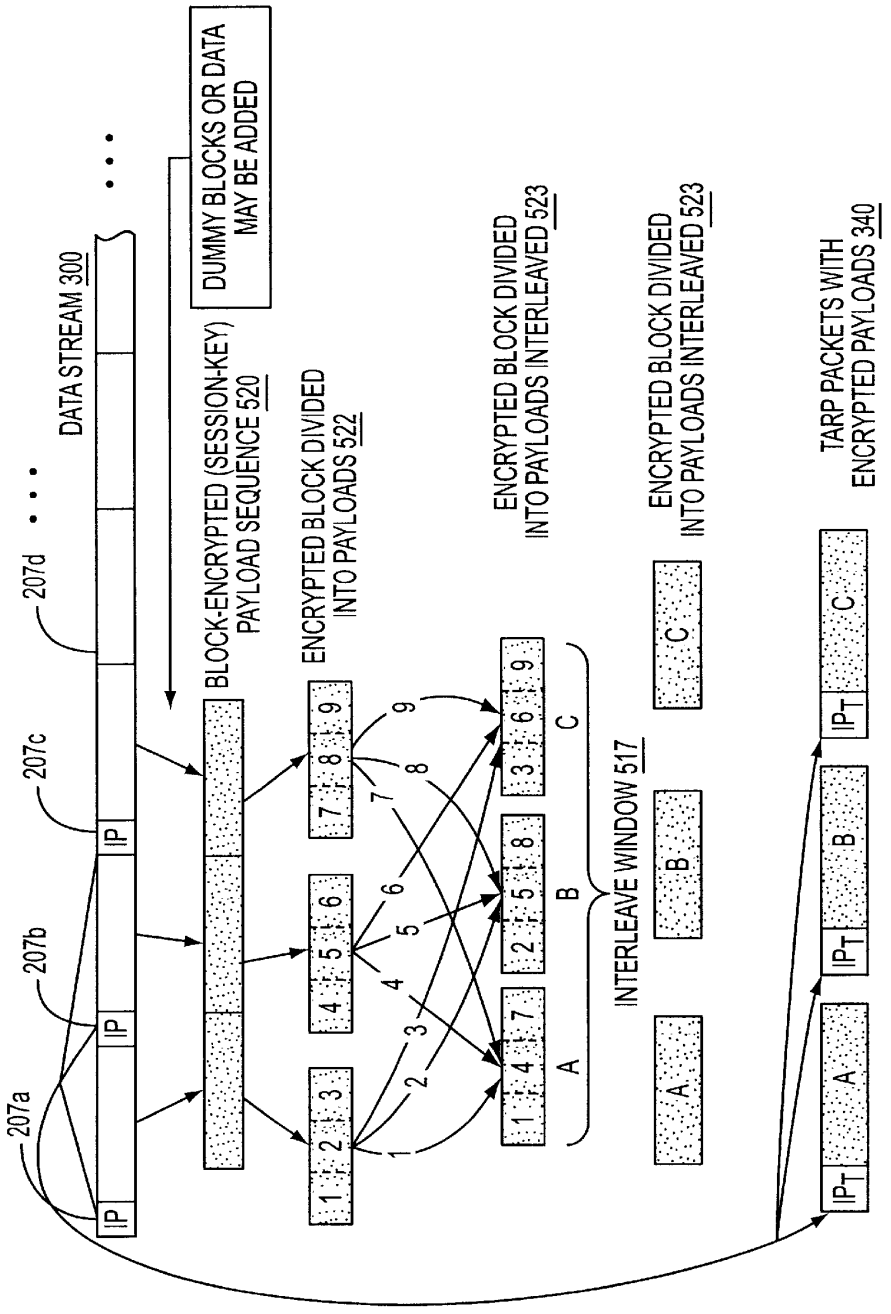

DATA STREAM 300
207a
207b
207c
207d
IP
IPT
DUMMY BLOCKS OR DATA MAY BE ADDED
BLOCK-ENCRYPTED (SESSION-KEY) PAYLOAD SEQUENCE 520
ENCRYPTED BLOCK DIVIDED INTO PAYLOADS 522
ENCRYPTED BLOCK DIVIDED INTO PAYLOADS INTERLEAVED 523
INTERLEAVE WINDOW 517
A
B
C
TARP PACKETS WITH ENCRYPTED PAYLOADS 340

FIG. 3B

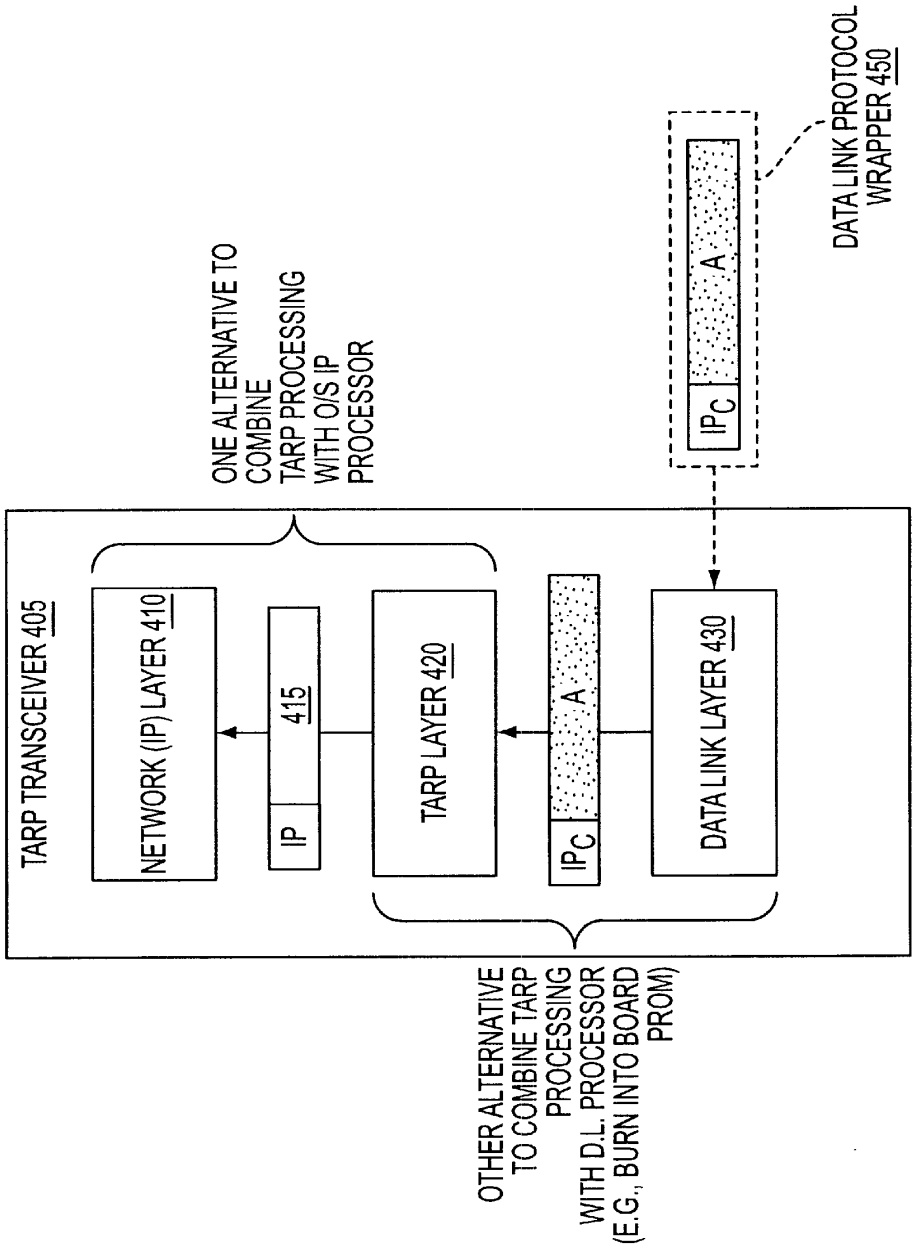

TARP TRANSCEIVER 405
NETWORK (IP) LAYER 410
IP 415
TARP LAYER 420
IPC A
DATALINK LAYER 430
ONE ALTERNATIVE TO COMBINE TARP PROCESSING WITH O/S IP PROCESSOR
IPC A
DATALINK PROTOCOL WRAPPER 450
OTHER ALTERNATIVE TO COMBINE TARP PROCESSING WITH D.L. PROCESSOR (E.G., BURN INTO BOARD PROM)

FIG. 4

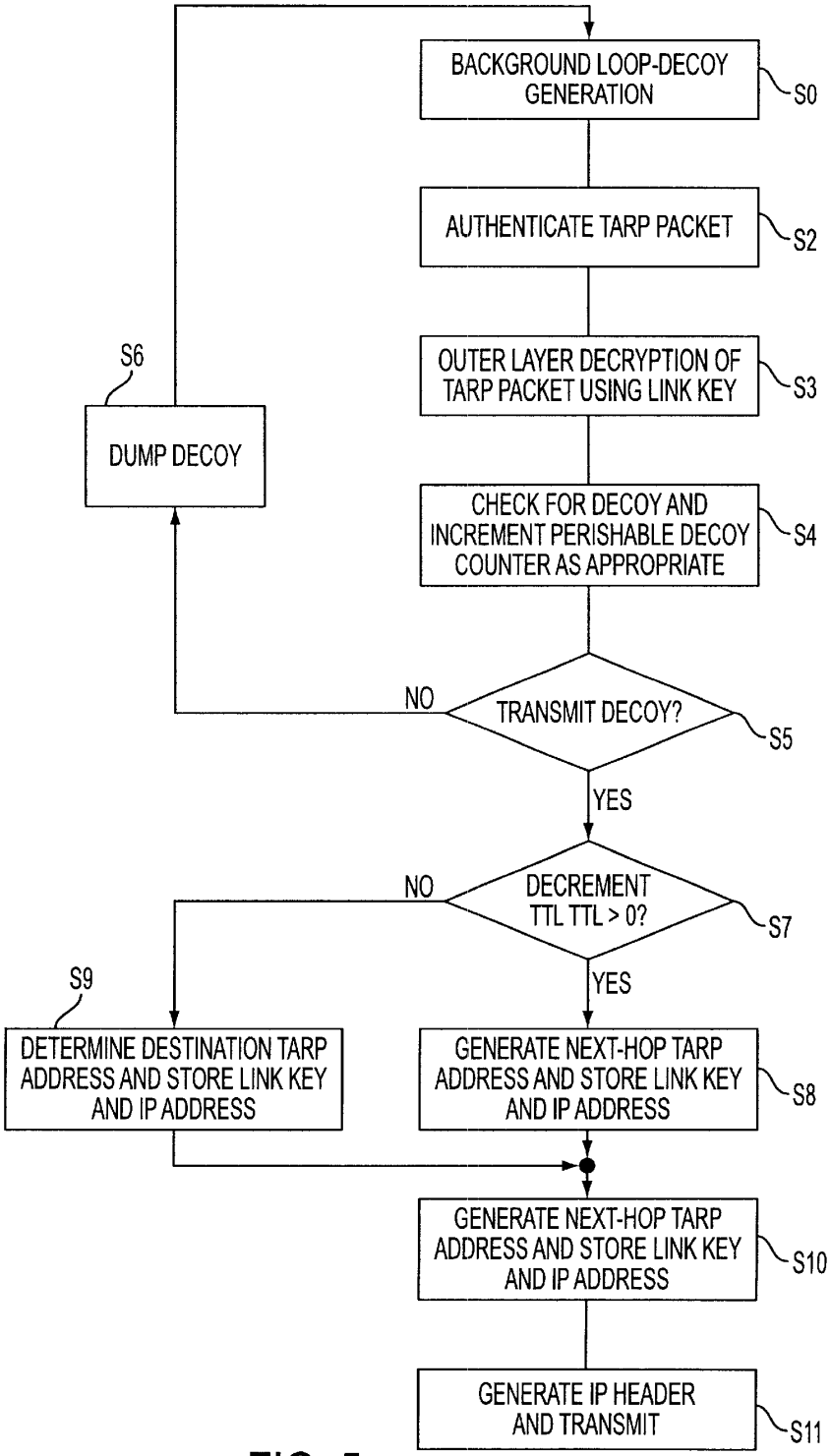

BACKGROUND LOOP-DECOY GENERATION
S0
AUTHENTICATE TARP PACKET
S2
OUTER LAYER DECRYPTION OF TARP PACKET USING LINK KEY
S3
CHECK FOR DECOY AND INCREMENT PERISHABLE DECOY COUNTER AS APPROPRIATE
S4
TRANSMIT DECOY?
S5
NO
YES
S6
DUMP DECOY
DECREMENT TTL TTL > 0?
S7
NO
YES
S9
DETERMINE DESTINATION TARP ADDRESS AND STORE LINK KEY AND IP ADDRESS
GENERATE NEXT-HOP TARP ADDRESS AND STORE LINK KEY AND IP ADDRESS
S8
GENERATE NEXT-HOP TARP ADDRESS AND STORE LINK KEY AND IP ADDRESS
S10
GENERATE IP HEADER AND TRANSMIT
S11

FIG. 5

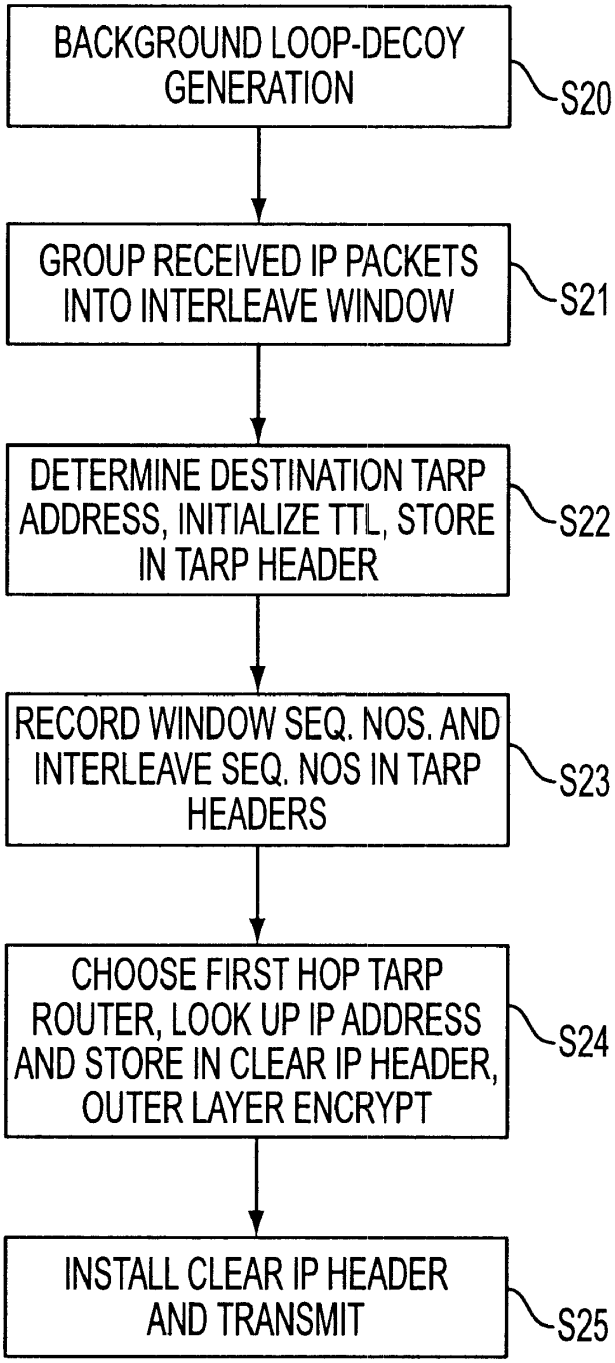

BACKGROUND LOOP-DECOY GENERATION
S20
GROUP RECEIVED IP PACKETS INTO INTERLEAVE WINDOW
S21
DETERMINE DESTINATION TARP ADDRESS, INITIALIZE TTL, STORE IN TARP HEADER
S22
RECORD WINDOW SEQ. NOS. AND INTERLEAVE SEQ. NOS IN TARP HEADERS
S23
CHOOSE FIRST HOP TARP ROUTER, LOOK UP IP ADDRESS AND STORE IN CLEAR IP HEADER, OUTER LAYER ENCRYPT
S24
INSTALL CLEAR IP HEADER AND TRANSMIT
S25

FIG. 6

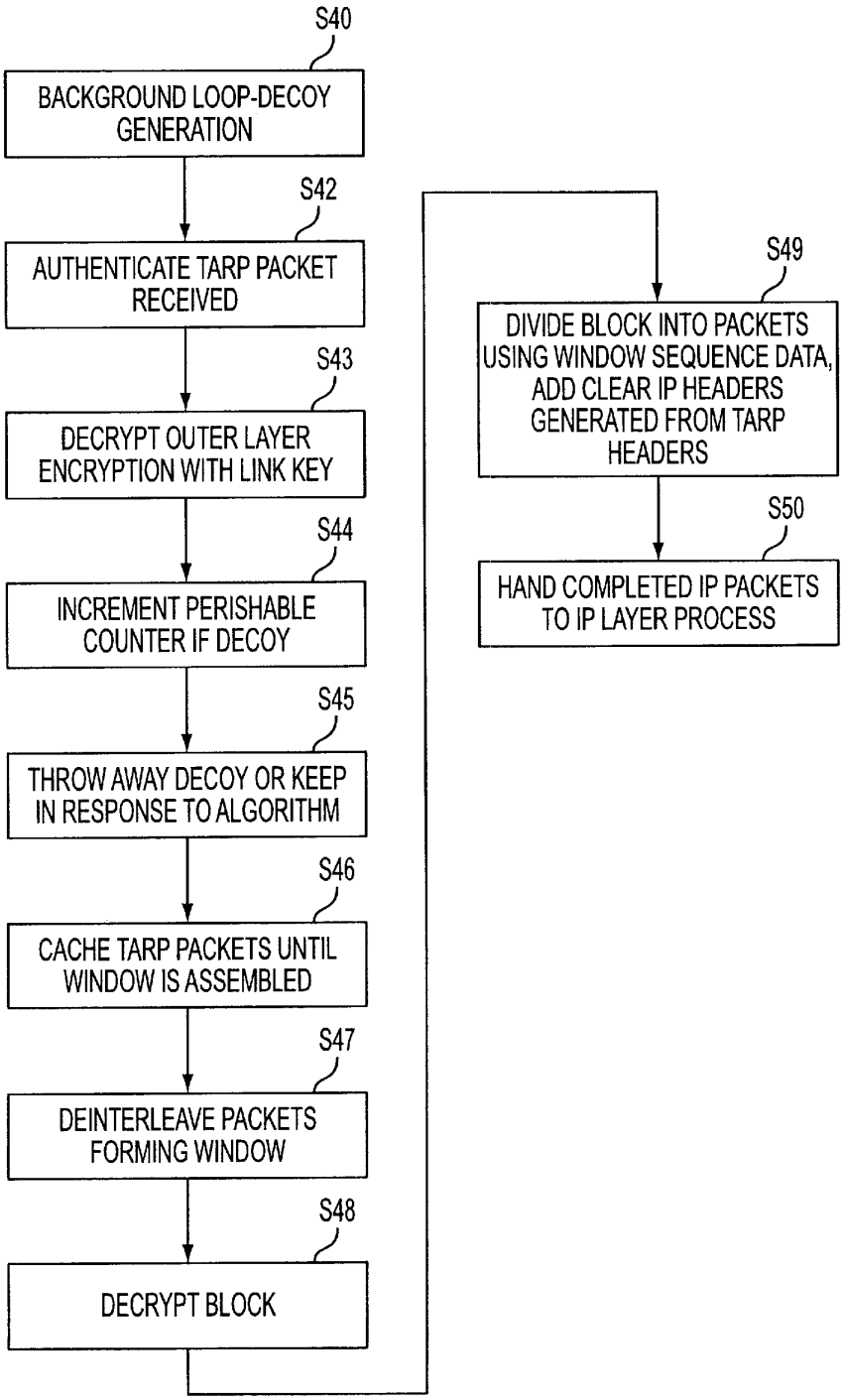

S40
BACKGROUND LOOP-DECOY GENERATION
S42
AUTHENTICATE TARP PACKET RECEIVED
S43
DECRYPT OUTER LAYER ENCRYPTION WITH LINK KEY
S44
INCREMENT PERISHABLE COUNTER IF DECOY
S45
THROW AWAY DECOY OR KEEP IN RESPONSE TO ALGORITHM
S46
CACHE TARP PACKETS UNTIL WINDOW IS ASSEMBLED
S47
DEINTERLEAVE PACKETS FORMING WINDOW
S48
DECRYPT BLOCK
S49
DIVIDE BLOCK INTO PACKETS USING WINDOW SEQUENCE DATA, ADD CLEAR IP HEADERS GENERATED FROM TARP HEADERS
S50
HAND COMPLETED IP PACKETS TO IP LAYER PROCESS

FIG. 7

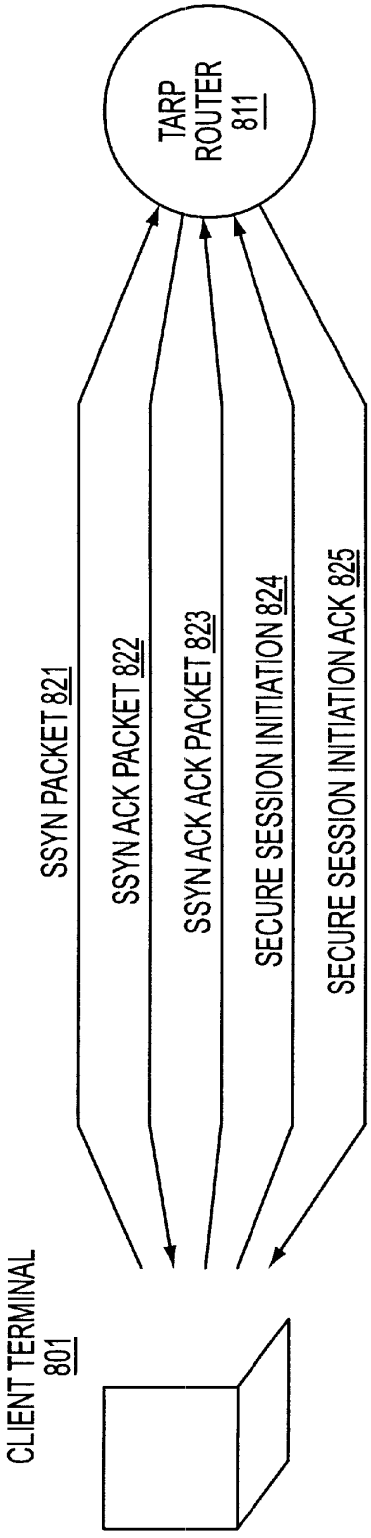
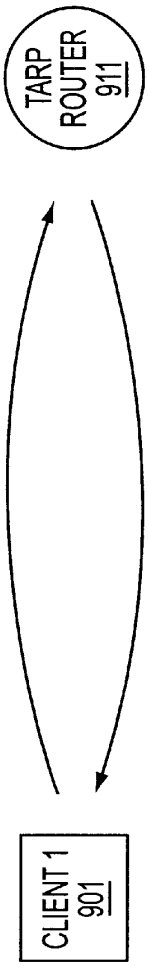

TARP ROUTER 811
SSYN PACKET 821
SSYN ACK PACKET 822
SSYN ACK ACK PACKET 823
SECURE SESSION INITIATION 824
SECURE SESSION INITIATION ACK 825
CLIENT TERMINAL 801

FIG. 8



TRANSMIT TABLE 921

131.218.204.98	,	131.218.204.65
131.218.204.221	,	131.218.204.97
131.218.204.139	,	131.218.204.186
131.218.204.12	,	131.218.204.55
.		.
.		.
.		.

RECEIVE TABLE 922

131.218.204.161	,	131.218.204.89
131.218.204.66	,	131.218.204.212
131.218.204.201	,	131.218.204.127
131.218.204.119	,	131.218.204.49
.		.
.		.
.		.

RECEIVE TABLE 924

131.218.204.98	,	131.218.204.65
131.218.204.221	,	131.218.204.97
131.218.204.139	,	131.218.204.186
131.218.204.12	,	131.218.204.55
.		.
.		.
.		.

TRANSMIT TABLE 923

131.218.204.161	,	131.218.204.89
131.218.204.66	,	131.218.204.212
131.218.204.201	,	131.218.204.127
131.218.204.119	,	131.218.204.49
.		.
.		.
.		.

FIG. 9

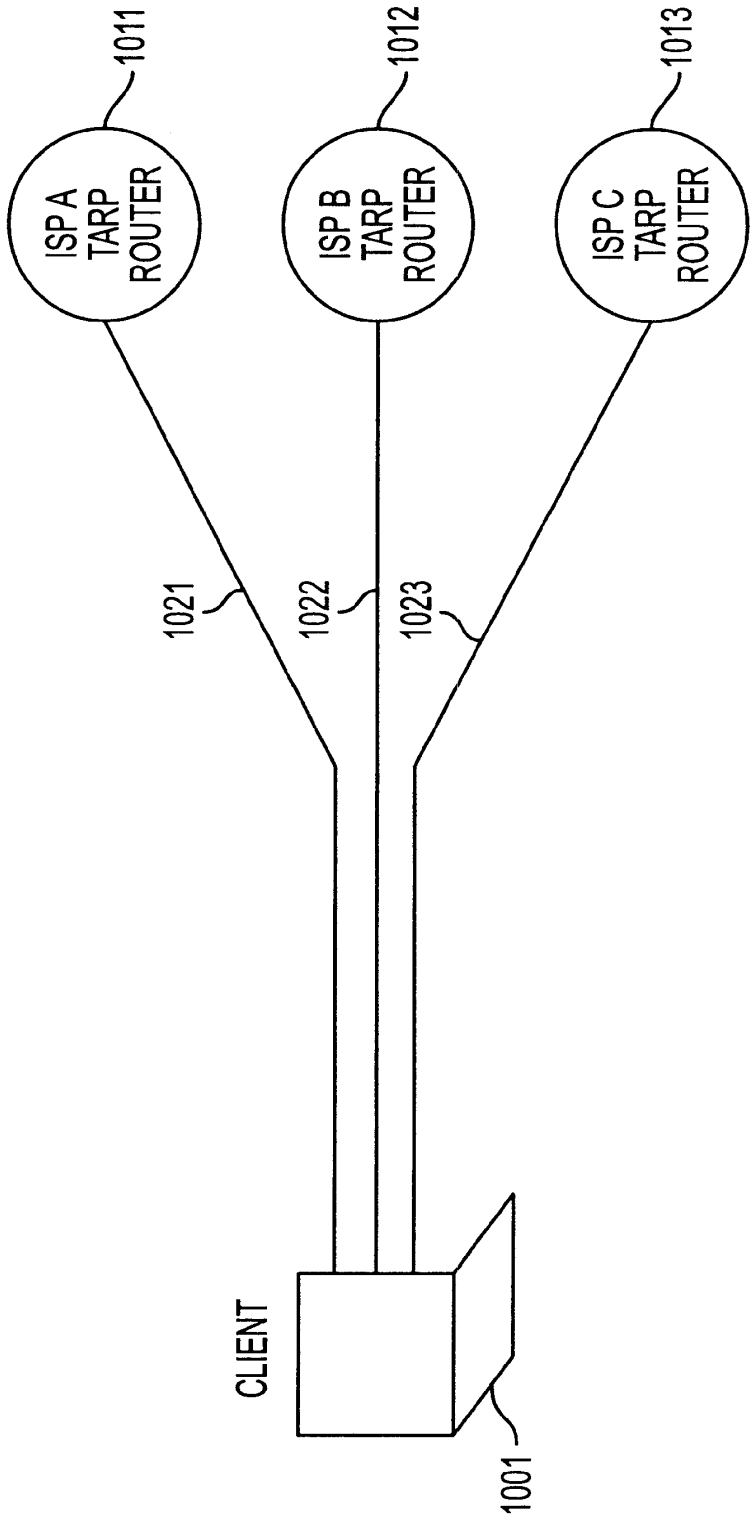

1011
1012
1013
ISP A TARP ROUTER
ISP B TARP ROUTER
ISP C TARP ROUTER
1021
1022
1023
CLIENT
1001

FIG. 10

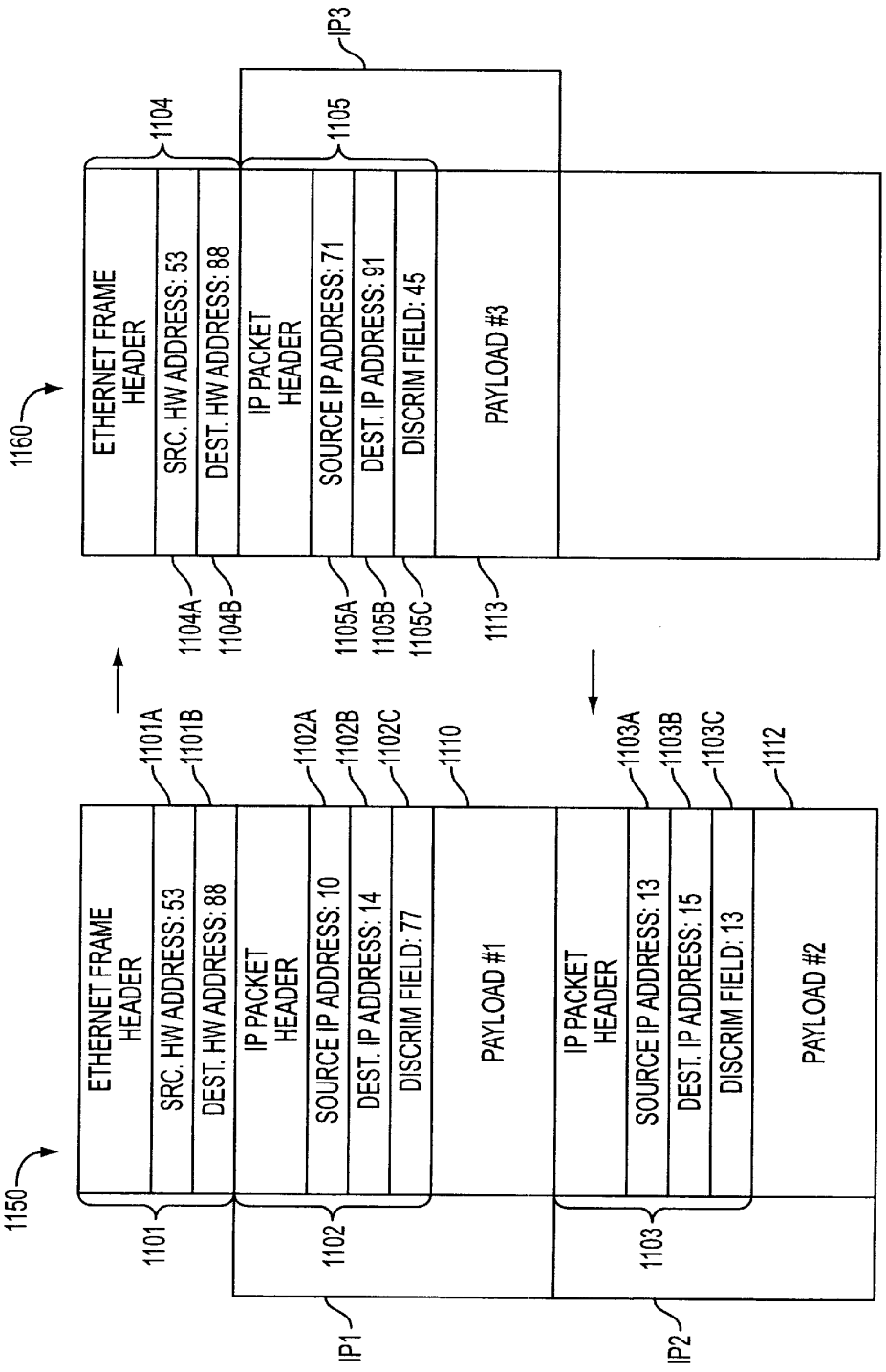

1160
ETHERNET FRAME HEADER
SRC. HW ADDRESS: 53
DEST. HW ADDRESS: 88
IP PACKET HEADER
SOURCE IP ADDRESS: 71
DEST. IP ADDRESS: 91
DISCRIM FIELD: 45
PAYLOAD #3
IP3
1104
1105
1104A
1104B
1105A
1105B
1105C
1113
1150
ETHERNET FRAME HEADER
SRC. HW ADDRESS: 53
DEST. HW ADDRESS: 88
IP PACKET HEADER
SOURCE IP ADDRESS: 10
DEST. IP ADDRESS: 14
DISCRIM FIELD: 77
PAYLOAD #1
IP PACKET HEADER
SOURCE IP ADDRESS: 13
DEST. IP ADDRESS: 15
DISCRIM FIELD: 13
PAYLOAD #2
1101A
1101B
1102A
1102B
1102C
1110
1103A
1103B
1103C
1112
1101
1102
1103
IP1
IP2

FIG. 11

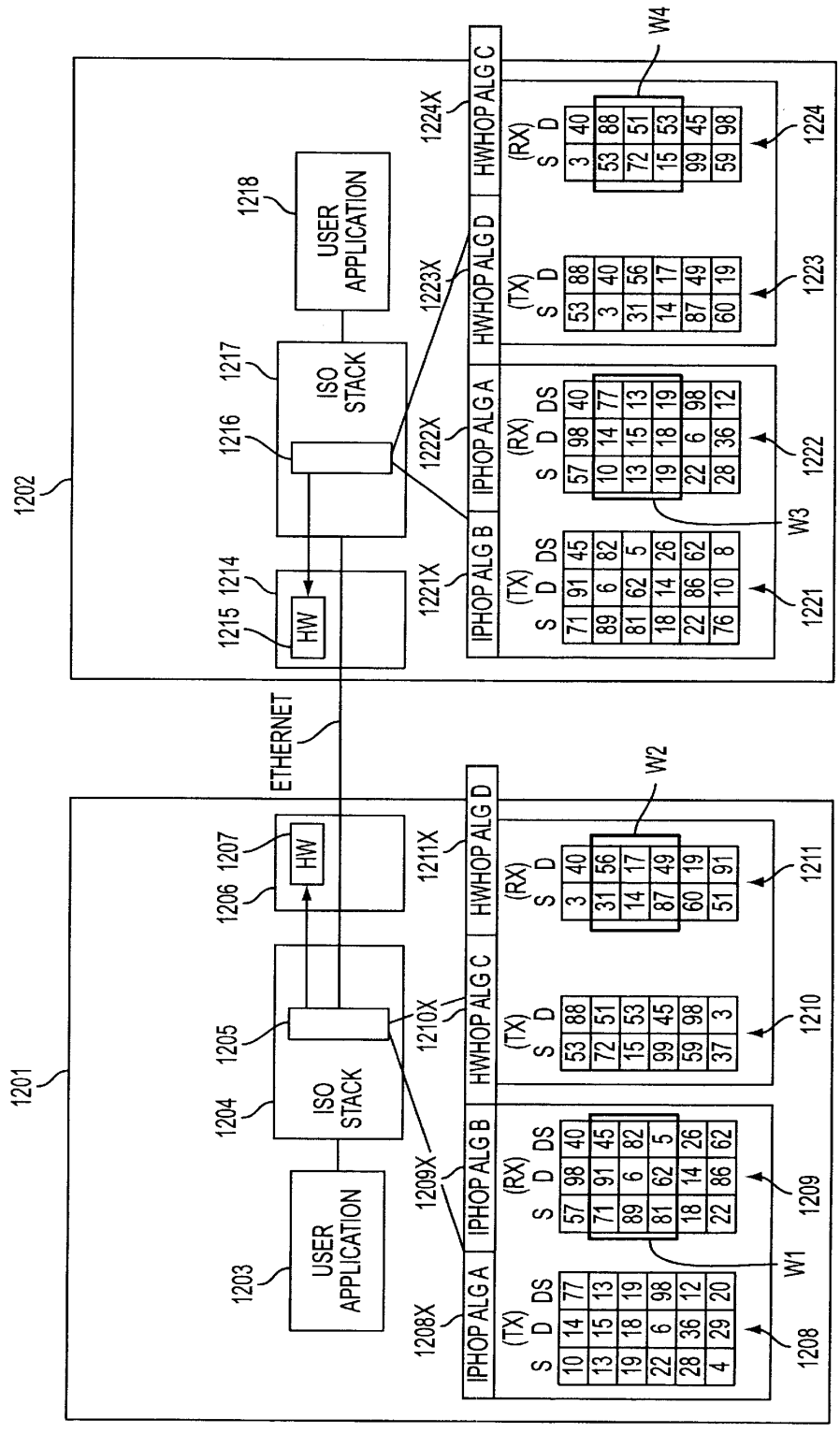

1201
1202
1203
USER APPLICATION
1204
ISO STACK
1205
1206
1207
HW
ETHERNET
1208X
IPHOP ALG A
1209X
IPHOP ALG B
1210X
HWHOP ALG C
1211X
HWHOP ALG D
1208
1209
1210
1211
W1
W2
1214
1215
1216
1217
ISO STACK
1218
USER APPLICATION
1221X
IPHOP ALG B
1222X
IPHOP ALG A
1223X
HWHOP ALG D
1224X
HWHOP ALG C
1221
1222
1223
1224
W3
W4
(TX)
(RX)
S
D
DS

FIG. 12A

MODE OR EMBODIMENT	HARDWARE ADDRESSES	IP ADDRESSES	DISCRIMINATOR FIELD VALUES
1. PROMISCUOUS	SAME FOR ALL NODES OR COMPLETELY RANDOM	CAN BE VARIED IN SYNC	CAN BE VARIED IN SYNC
2. PROMISCUOUS PER VPN	FIXED FOR EACH VPN	CAN BE VARIED IN SYNC	CAN BE VARIED IN SYNC
3. HARDWARE HOPPING	CAN BE VARIED IN SYNC	CAN BE VARIED IN SYNC	CAN BE VARIED IN SYNC

FIG. 12B

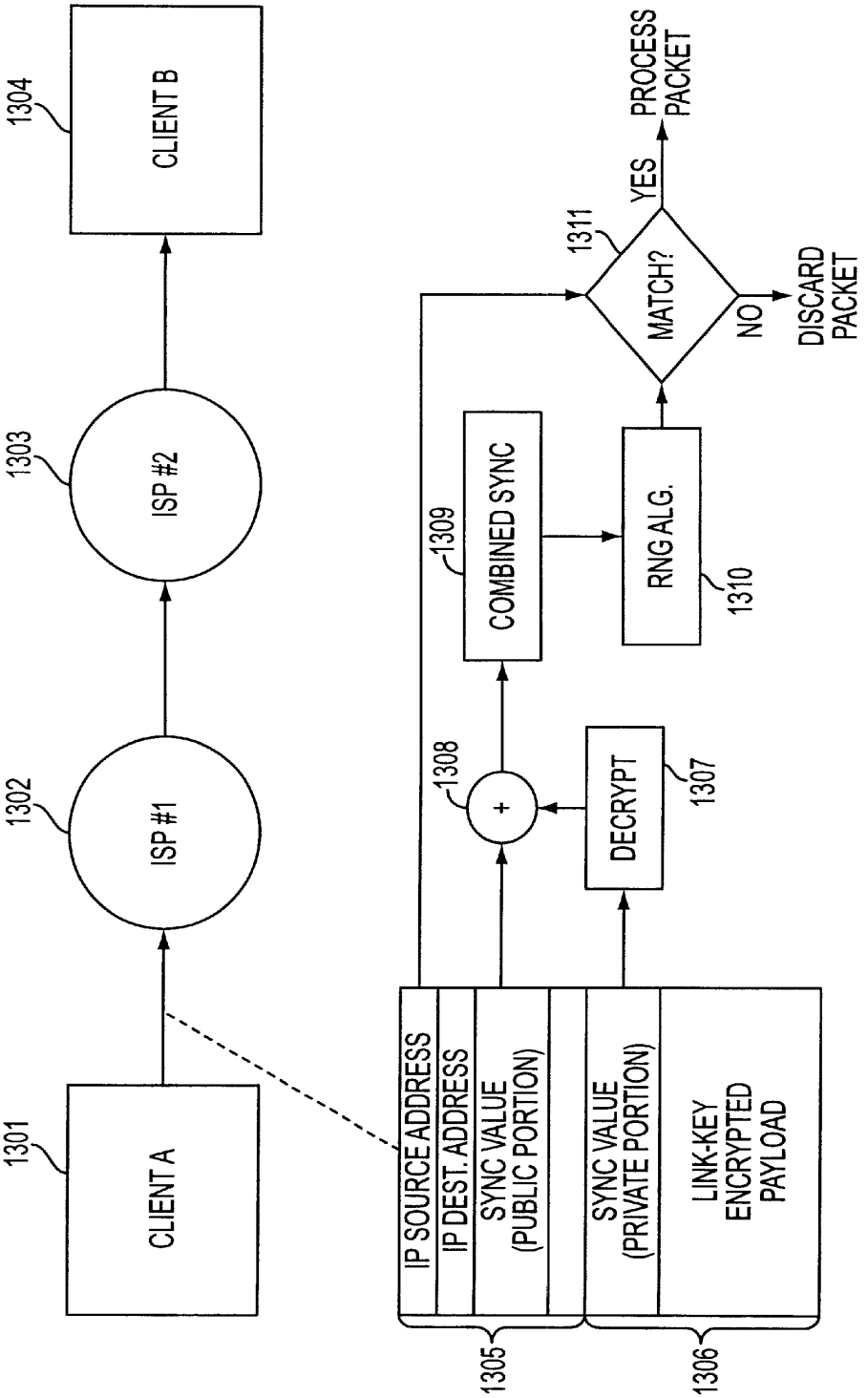

1301
CLIENT A
1302
ISP #1
1303
ISP #2
1304
CLIENT B
IP SOURCE ADDRESS
IP DEST. ADDRESS
SYNC VALUE (PUBLIC PORTION)
SYNC VALUE (PRIVATE PORTION)
LINK-KEY ENCRYPTED PAYLOAD
1305
1306
1307
DECRYPT
1308
+
1309
COMBINED SYNC
1310
RNG ALG.
1311
MATCH?
YES
PROCESS PACKET
NO
DISCARD PACKET

FIG. 13

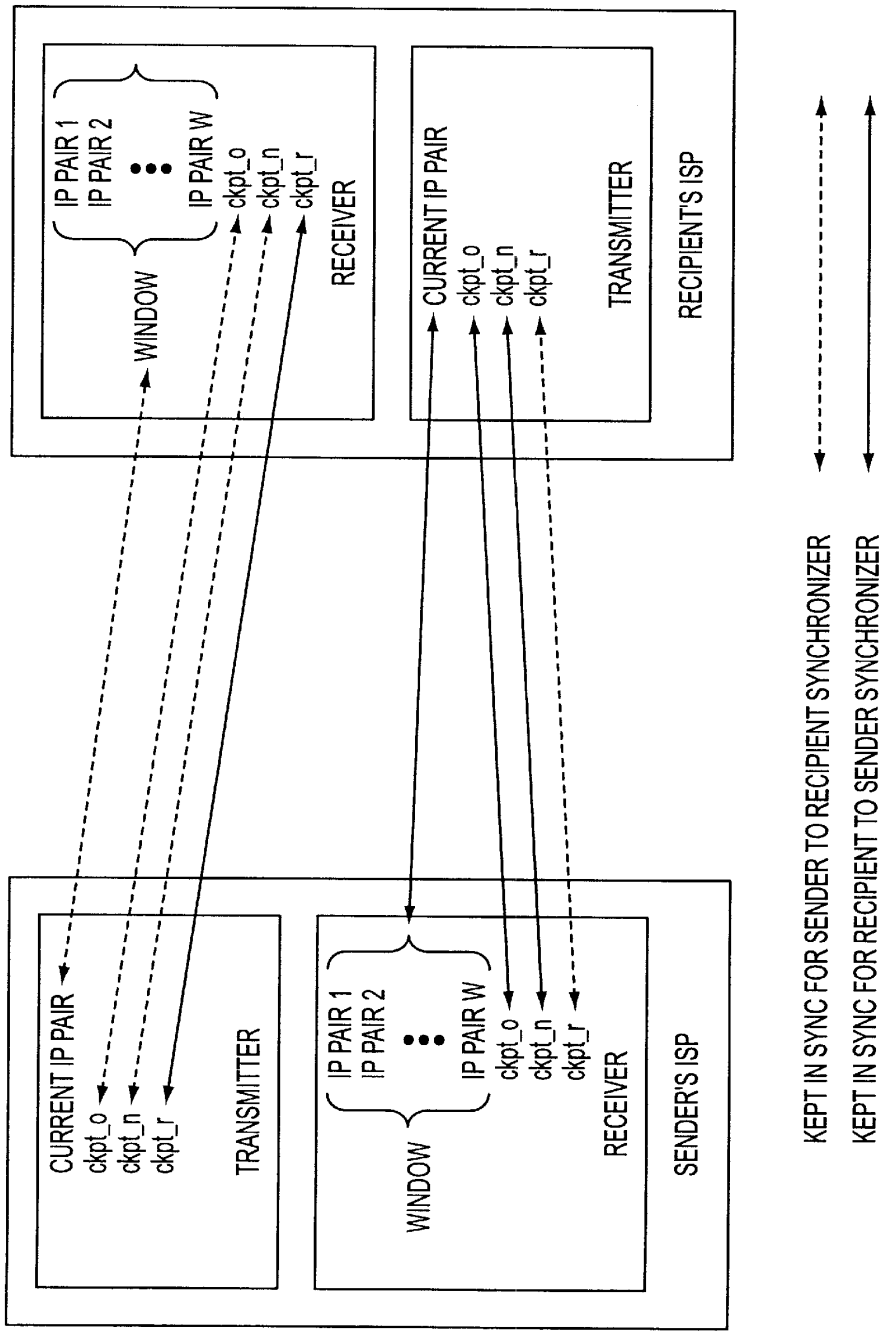

RECIPIENT'S ISP
TRANSMITTER
CURRENT IP PAIR
ckpt_o
ckpt_n
ckpt_r
RECEIVER
IP PAIR 1
IP PAIR 2
IP PAIR W
WINDOW
ckpt_o
ckpt_n
ckpt_r
SENDER'S ISP
TRANSMITTER
CURRENT IP PAIR
ckpt_o
ckpt_n
ckpt_r
RECEIVER
IP PAIR 1
IP PAIR 2
IP PAIR W
WINDOW
ckpt_o
ckpt_n
ckpt_r
KEPT IN SYNC FOR SENDER TO RECIPIENT SYNCHRONIZER
KEPT IN SYNC FOR RECIPIENT TO SENDER SYNCHRONIZER

FIG. 14

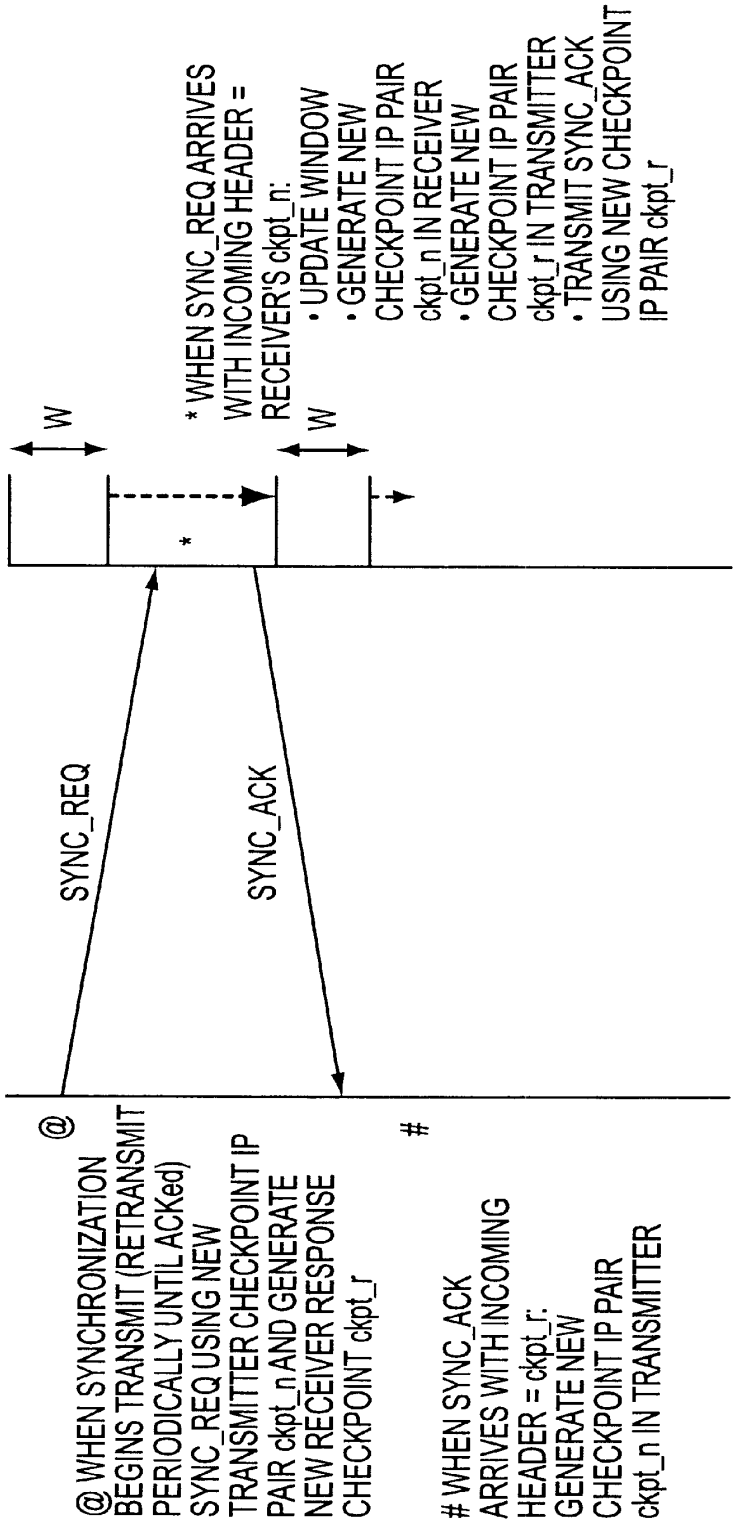

W
* WHEN SYNC_REQ ARRIVES WITH INCOMING HEADER = RECEIVER'S ckpt_n:
• UPDATE WINDOW
• GENERATE NEW CHECKPOINT IP PAIR ckpt_n IN RECEIVER
• GENERATE NEW CHECKPOINT IP PAIR ckpt_r IN TRANSMITTER
• TRANSMIT SYNC_ACK USING NEW CHECKPOINT IP PAIR ckpt_r
W
*
SYNC_REQ
SYNC_ACK
@
@ WHEN SYNCHRONIZATION BEGINS TRANSMIT (RETRANSMIT PERIODICALLY UNTIL ACKed) SYNC_REQ USING NEW TRANSMITTER CHECKPOINT IP PAIR ckpt_n AND GENERATE NEW RECEIVER RESPONSE CHECKPOINT ckpt_r
#
WHEN SYNC_ACK ARRIVES WITH INCOMING HEADER = ckpt_r: GENERATE NEW CHECKPOINT IP PAIR ckpt_n IN TRANSMITTER

FIG. 15

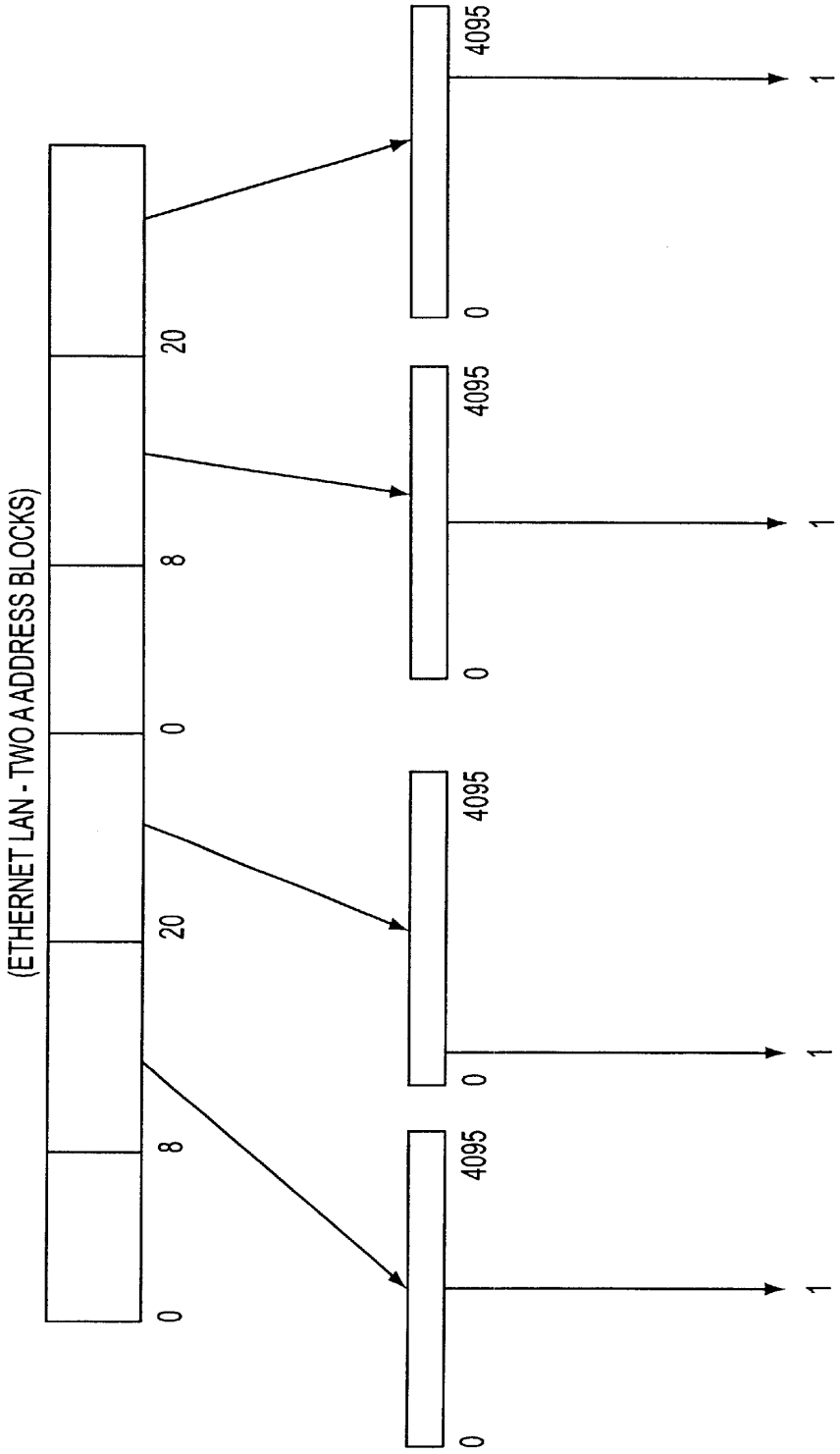

(ETHERNET LAN - TWO ADDRESS BLOCKS)
20
8
0
20
8
0
4095
0
4095
0
4095
0
4095
0
1
1
1
1

FIG. 16

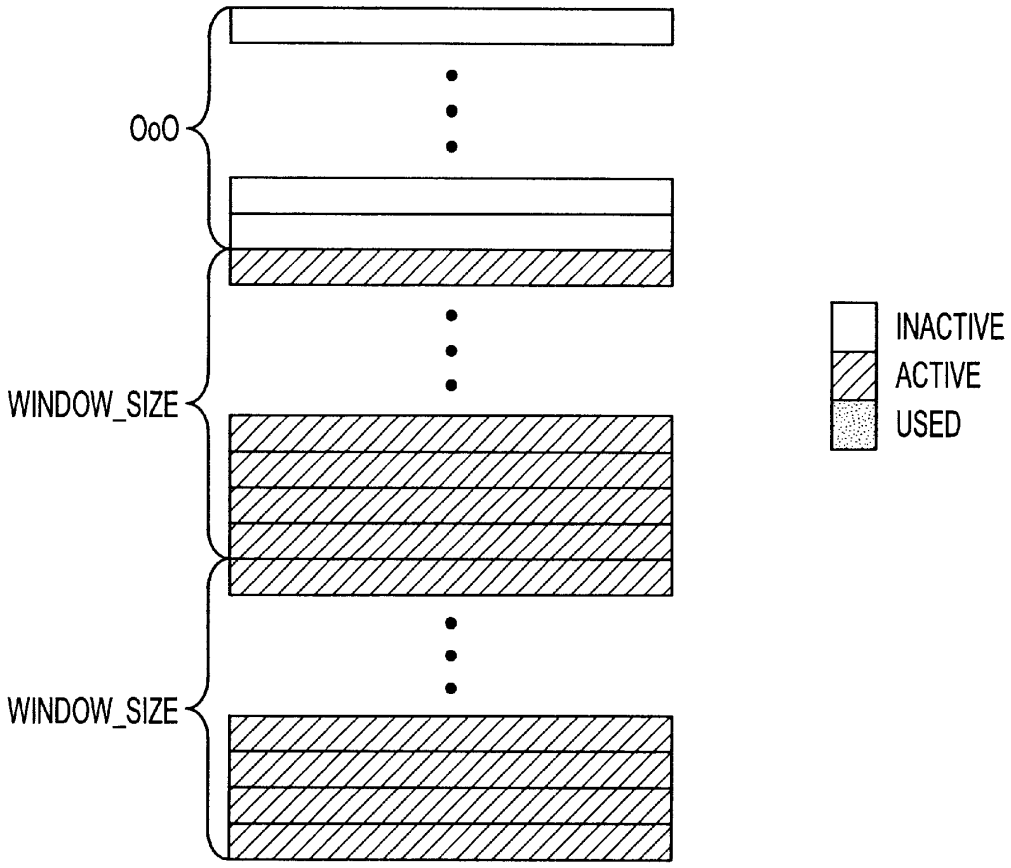


FIG. 17

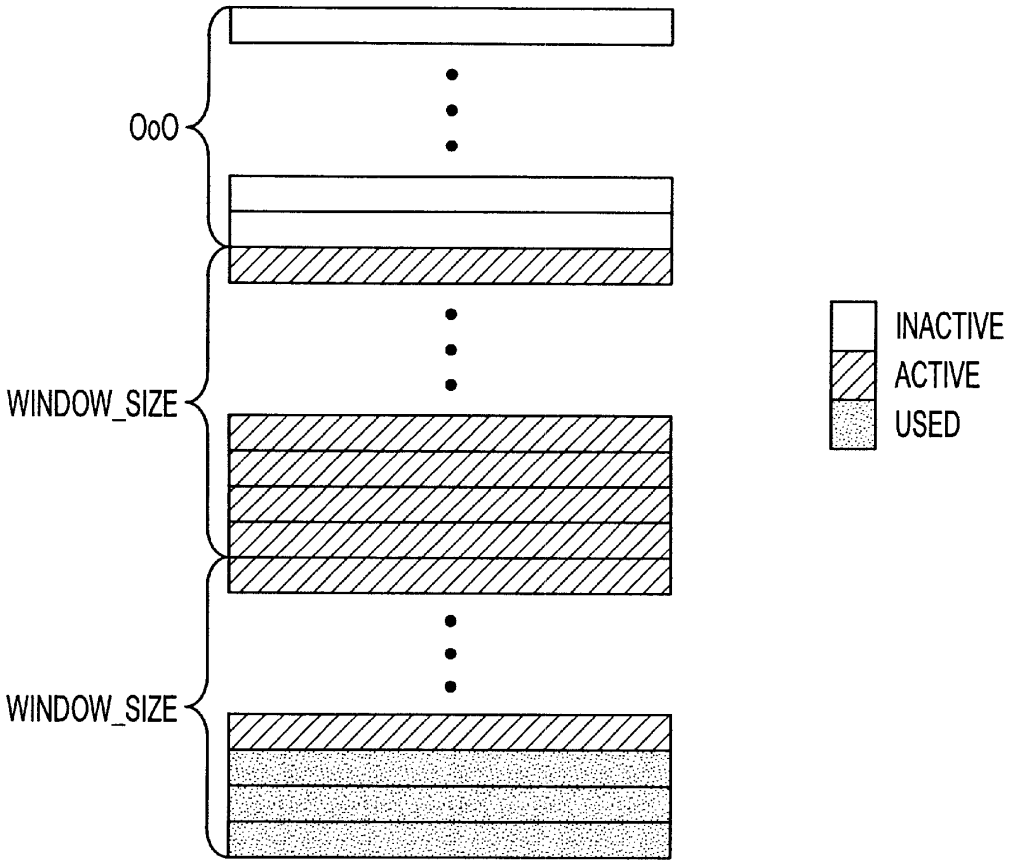


FIG. 18

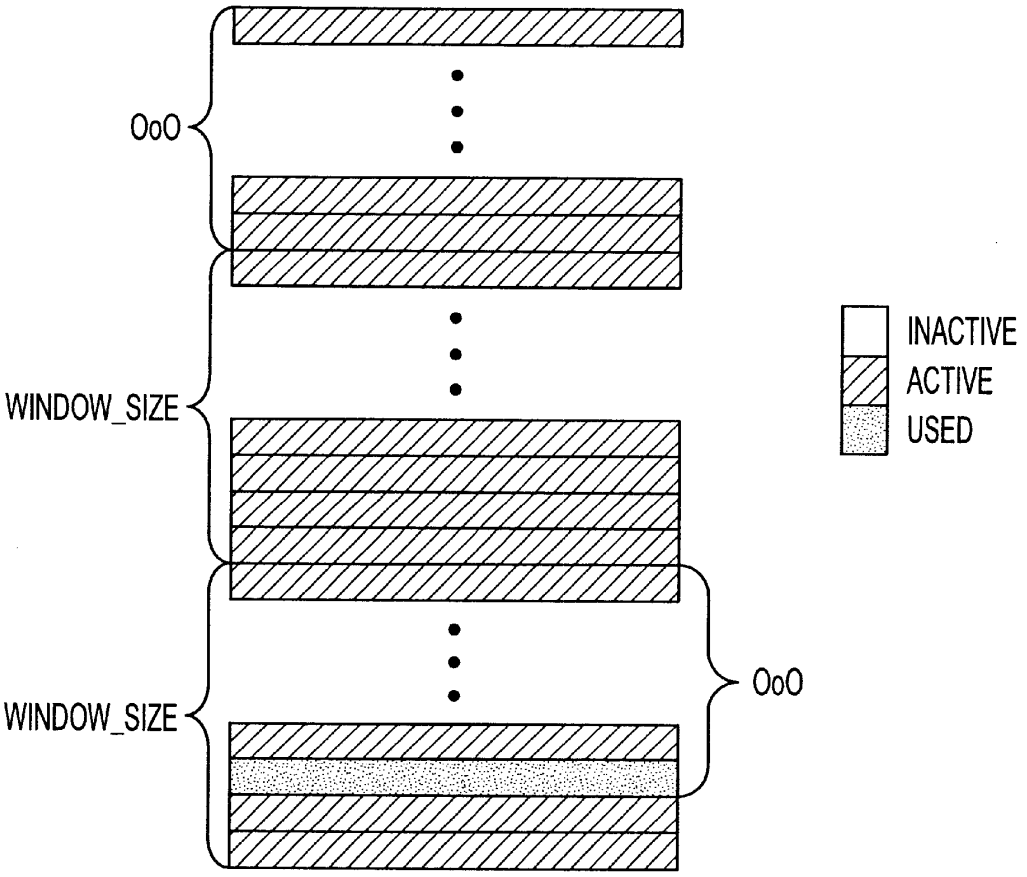


FIG. 19

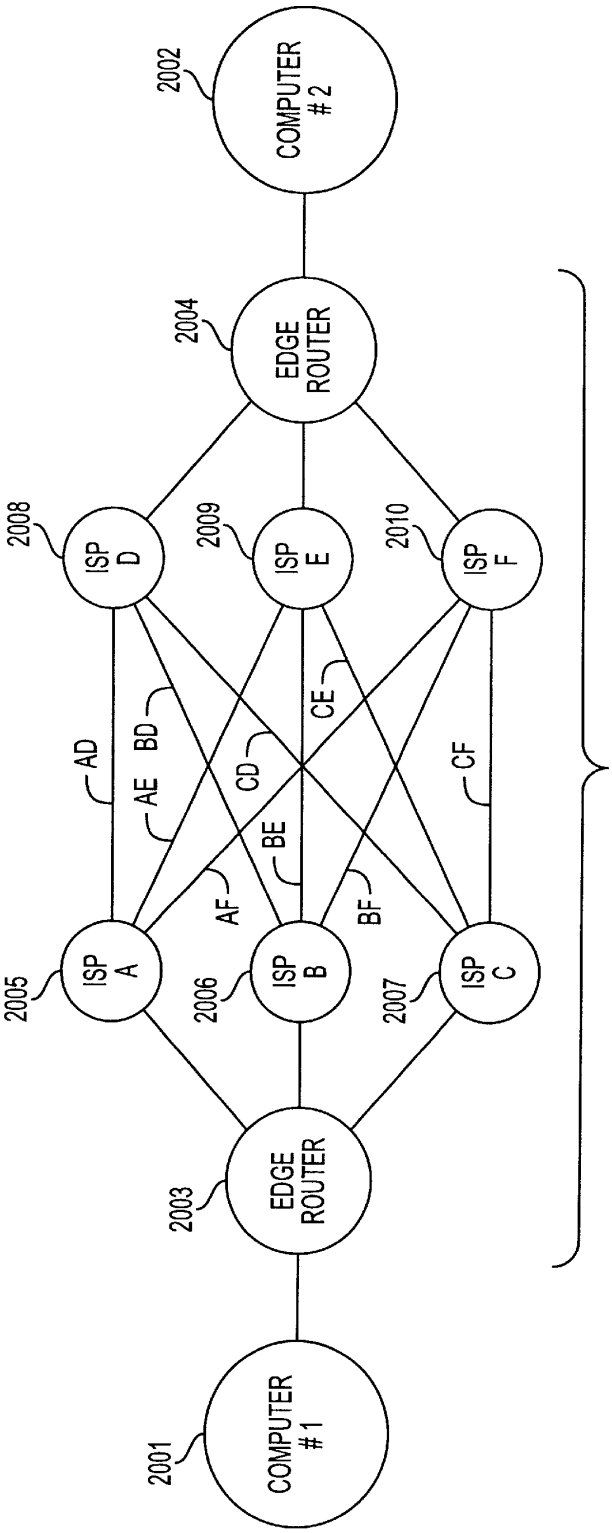

2002
COMPUTER #2
2004
EDGE ROUTER
2008
ISP D
2009
ISP E
2010
ISP F
AD
AE
BD
CD
CE
CF
AF
BE
BF
2005
ISP A
2006
ISP B
2007
ISP C
2003
EDGE ROUTER
2001
COMPUTER #1
2011

FIG. 20

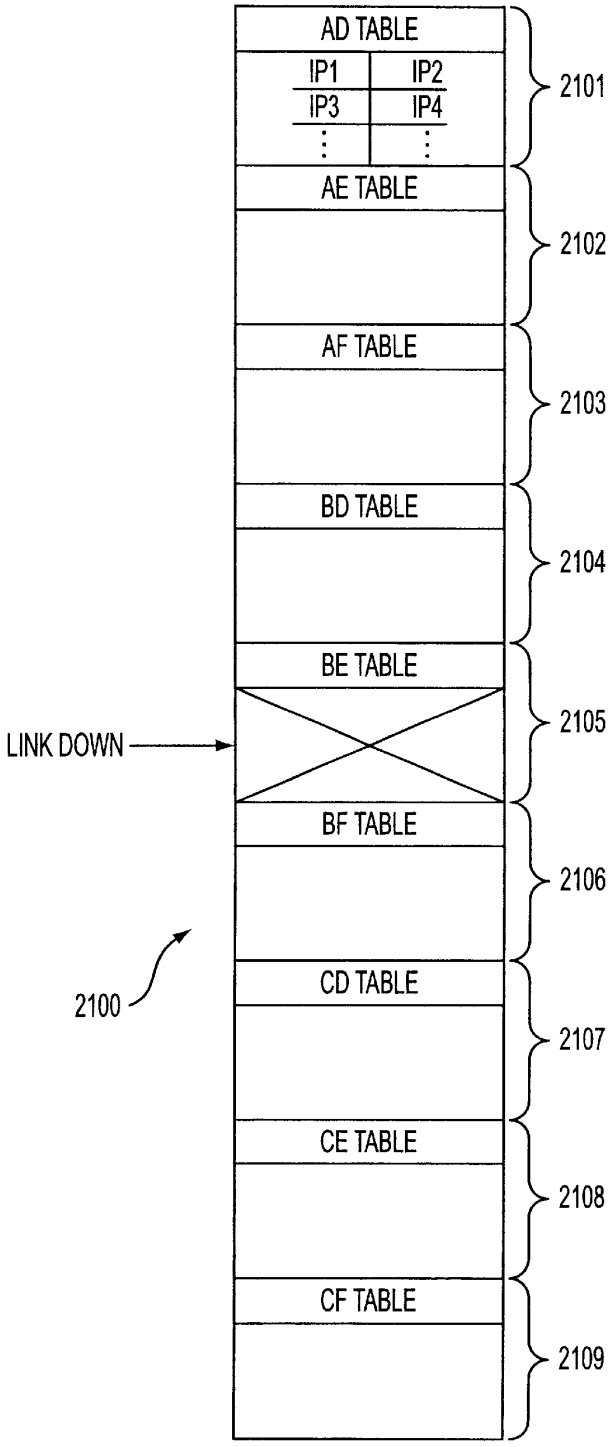

AD TABLE
IP1
IP2
IP3
IP4
2101
AE TABLE
2102
AF TABLE
2103
BD TABLE
2104
BE TABLE
LINK DOWN
2105
BF TABLE
2106
2100
CD TABLE
2107
CE TABLE
2108
CF TABLE
2109

FIG. 21

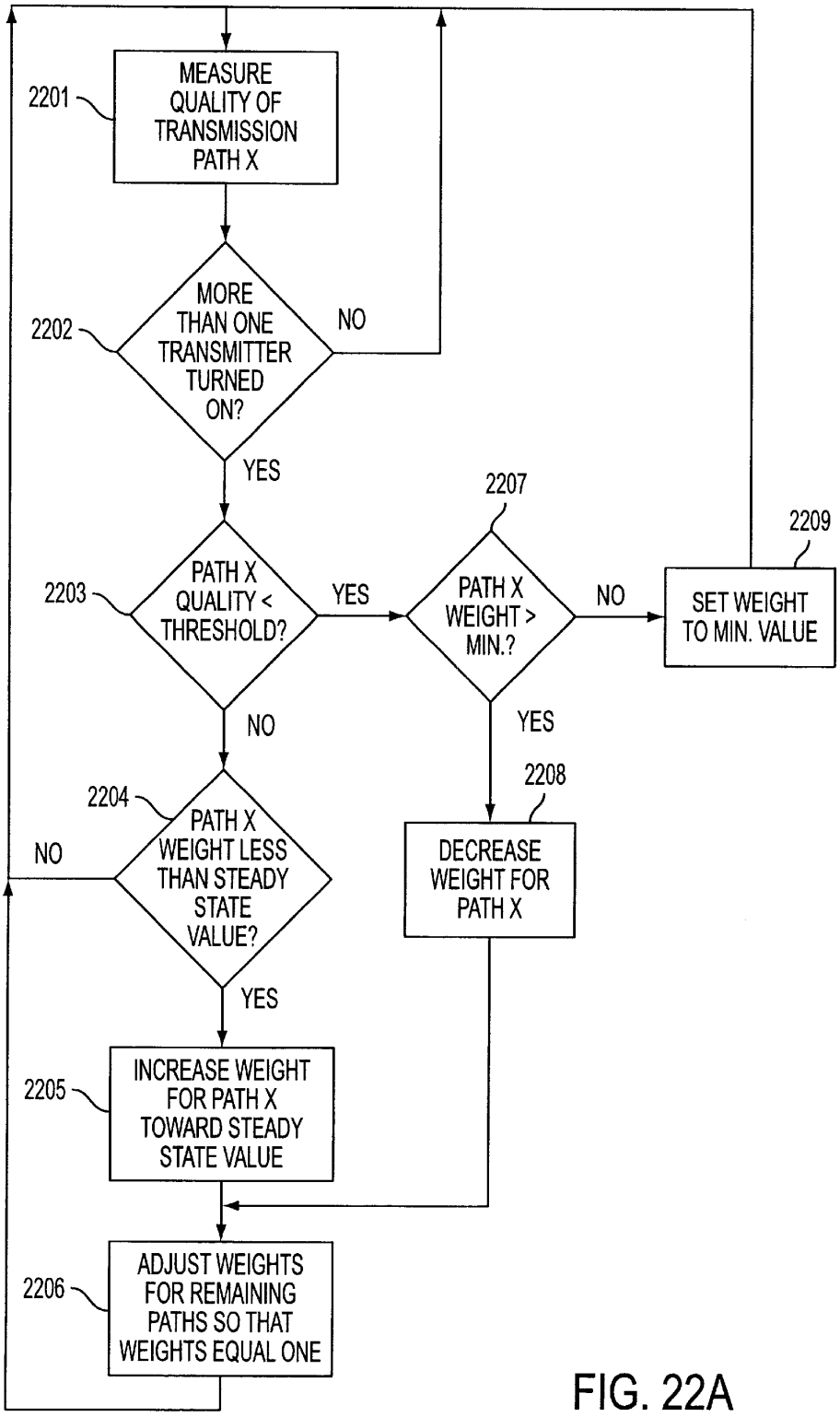

2201
MEASURE QUALITY OF TRANSMISSION PATH X
2202
MORE THAN ONE TRANSMITTER TURNED ON?
NO
YES
2203
PATH X QUALITY < THRESHOLD?
YES
2207
PATH X WEIGHT > MIN.?
NO
2209
SET WEIGHT TO MIN. VALUE
NO
YES
2204
PATH X WEIGHT LESS THAN STEADY STATE VALUE?
NO
2208
DECREASE WEIGHT FOR PATH X
YES
2205
INCREASE WEIGHT FOR PATH X TOWARD STEADY STATE VALUE
2206
ADJUST WEIGHTS FOR REMAINING PATHS SO THAT WEIGHTS EQUAL ONE

FIG. 22A

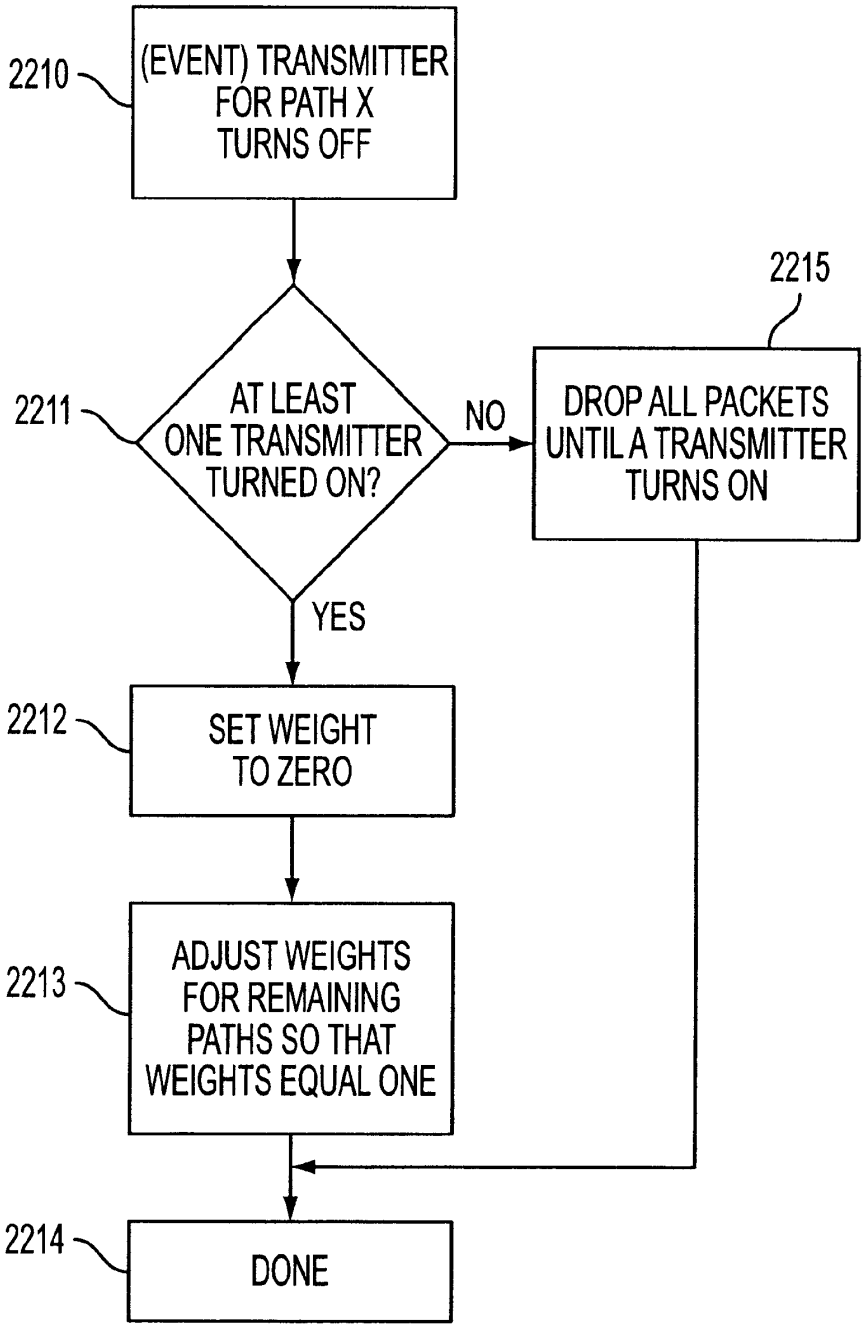

2210
(EVENT) TRANSMITTER FOR PATH X TURNS OFF
2211
AT LEAST ONE TRANSMITTER TURNED ON?
NO
2215
DROP ALL PACKETS UNTIL A TRANSMITTER TURNS ON
YES
2212
SET WEIGHT TO ZERO
2213
ADJUST WEIGHTS FOR REMAINING PATHS SO THAT WEIGHTS EQUAL ONE
2214
DONE

FIG. 22B

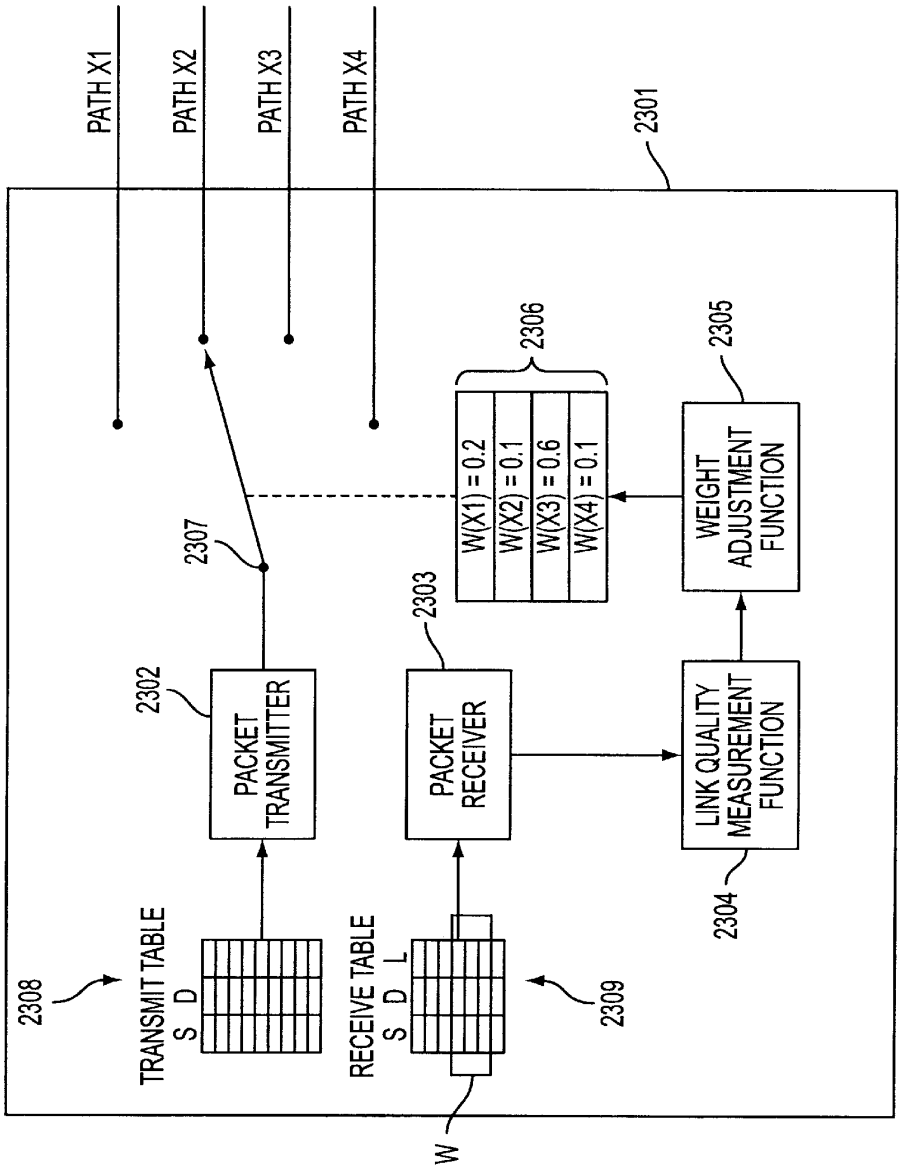

PATH X1
PATH X2
PATH X3
PATH X4
2301
2306
2305
W(X1) = 0.2
W(X2) = 0.1
W(X3) = 0.6
W(X4) = 0.1
WEIGHT ADJUSTMENT FUNCTION
2307
2303
2302
PACKET TRANSMITTER
PACKET RECEIVER
LINK QUALITY MEASUREMENT FUNCTION
2304
2308
TRANSMIT TABLE
S D
RECEIVE TABLE
S D L
2309
W

FIG. 23

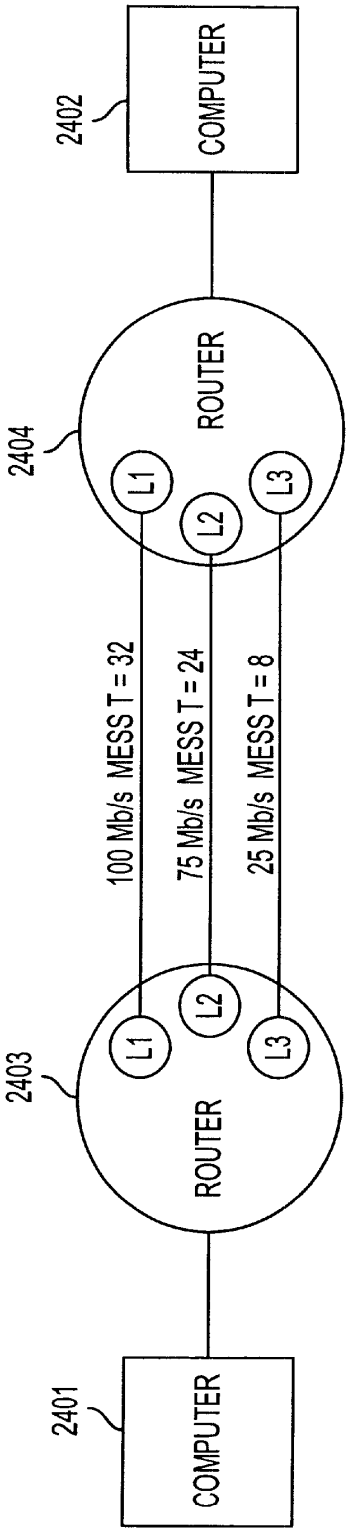

2402
COMPUTER
2404
ROUTER
L1
L2
L3
100 Mb/s MESS T = 32
75 Mb/s MESS T = 24
25 Mb/s MESS T = 8
2403
ROUTER
L1
L2
L3
2401
COMPUTER

FIG. 24

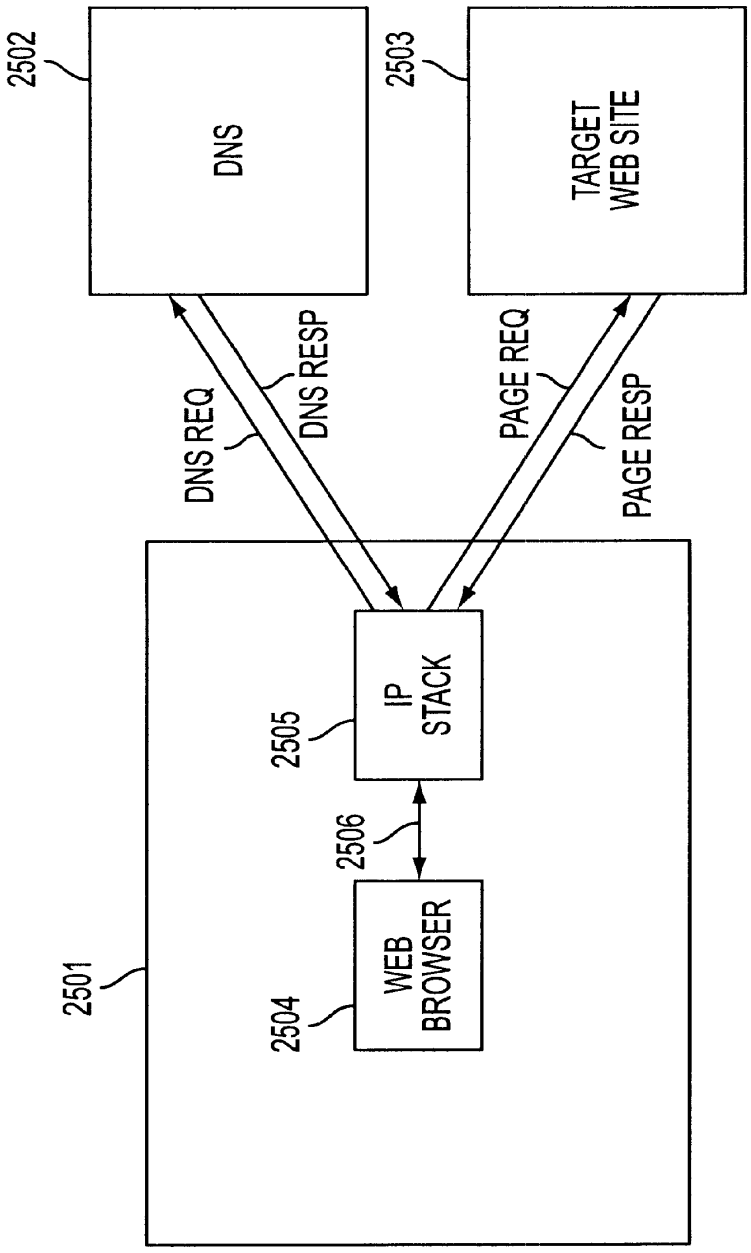


FIG. 25
(PRIOR ART)

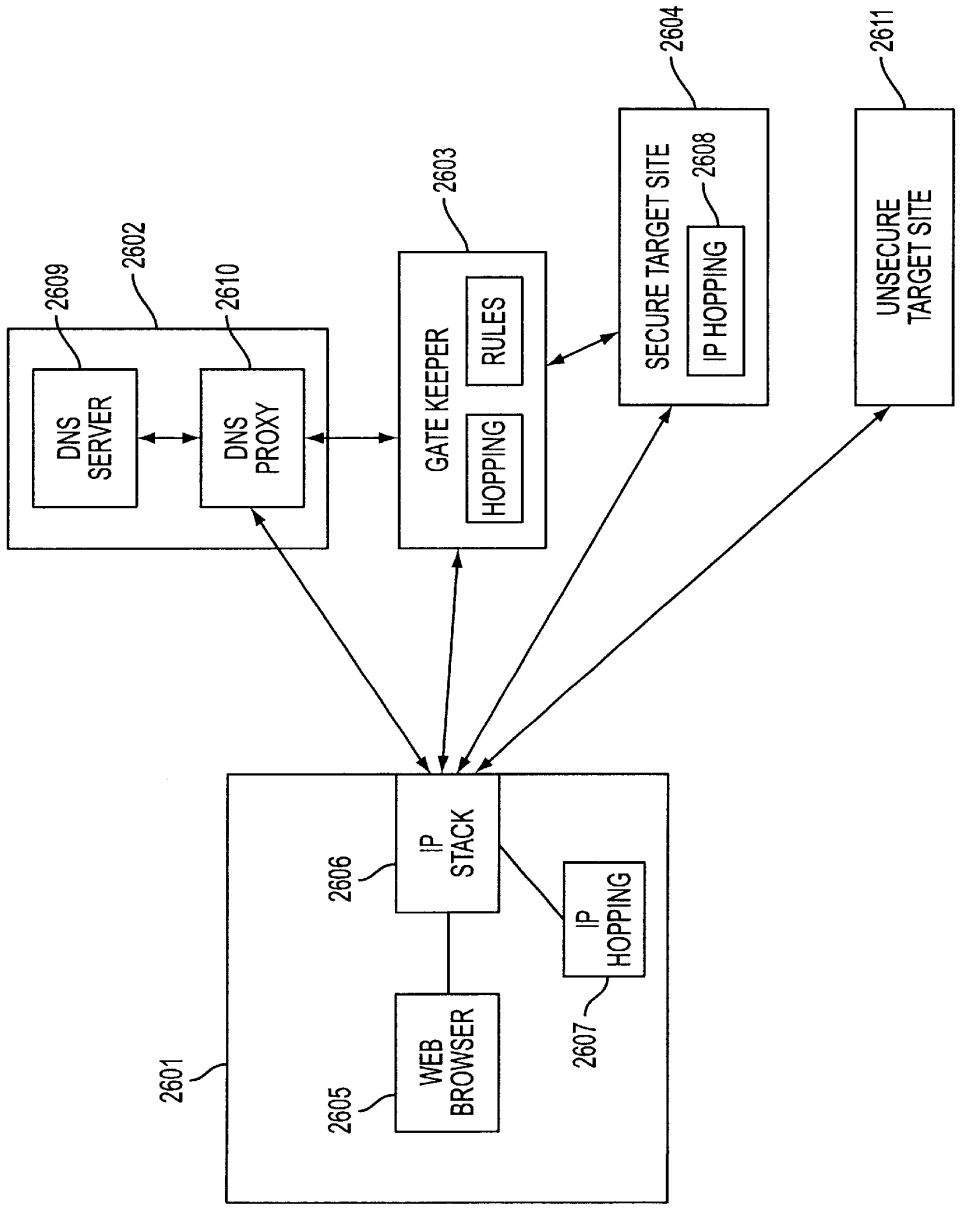

2601
2602
2603
2604
2605
2606
2607
2608
2609
2610
2611
DNS SERVER
DNS PROXY
GATE KEEPER
RULES
HOPPING
SECURE TARGET SITE
IP HOPPING
UNSECURE TARGET SITE
IP STACK
IP HOPPING
WEB BROWSER

FIG. 26

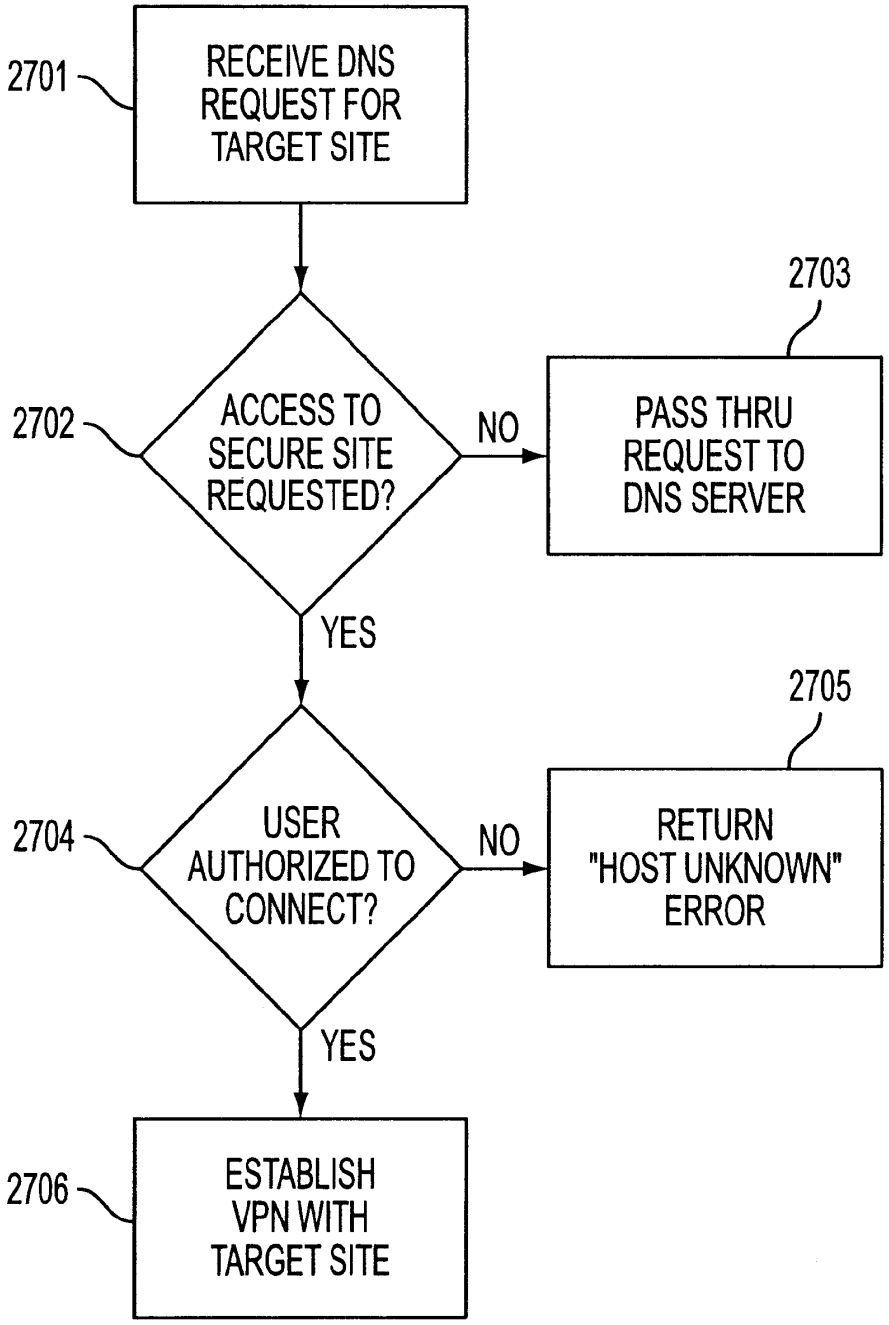

2701
RECEIVE DNS REQUEST FOR TARGET SITE
2702
ACCESS TO SECURE SITE REQUESTED?
NO
2703
PASS THRU REQUEST TO DNS SERVER
YES
2704
USER AUTHORIZED TO CONNECT?
NO
2705
RETURN "HOST UNKNOWN" ERROR
YES
2706
ESTABLISH VPN WITH TARGET SITE

FIG. 27

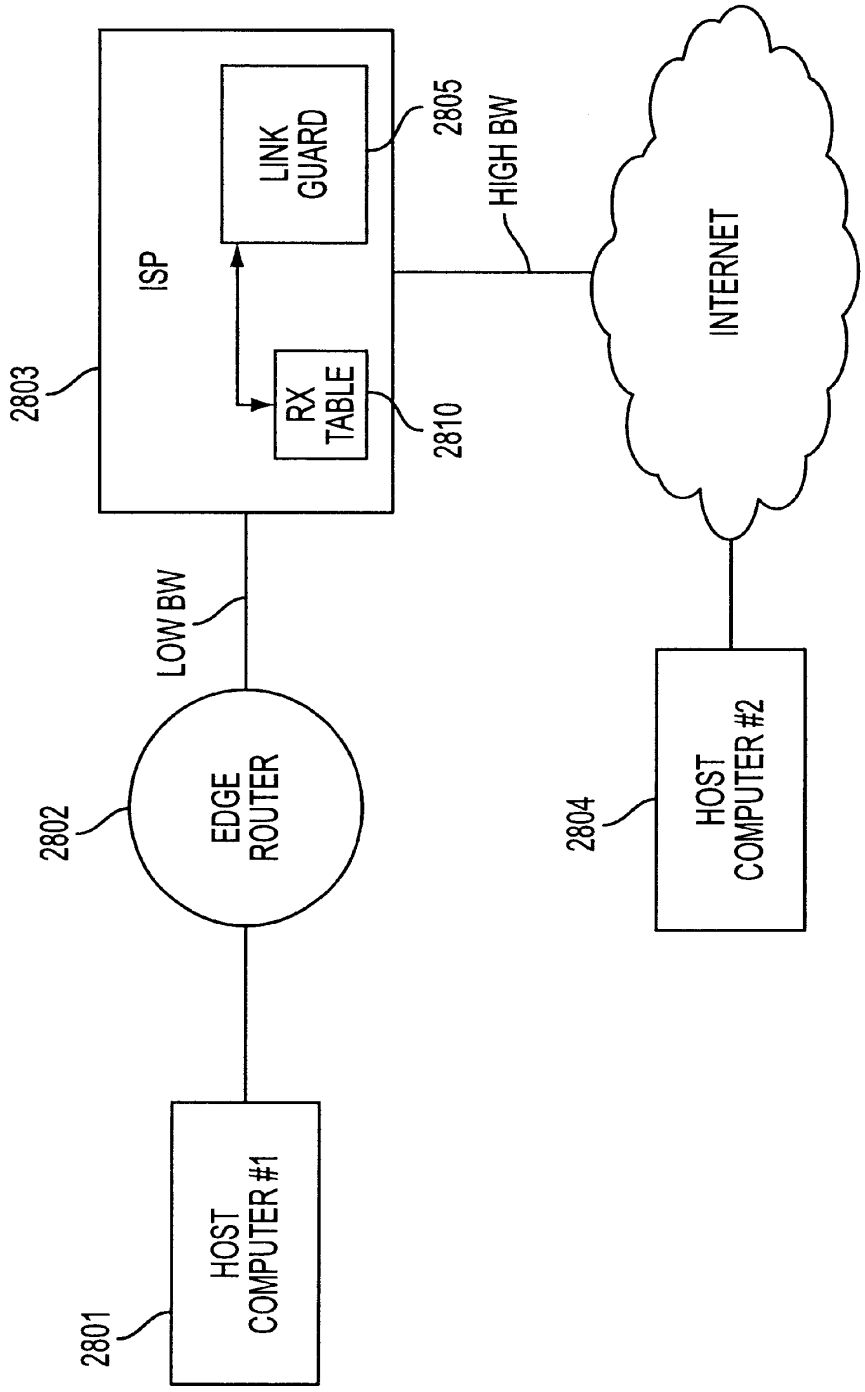

ISP
LINK GUARD
2805
HIGH BW
INTERNET
2803
RX TABLE
2810
LOW BW
EDGE ROUTER
2802
HOST COMPUTER #2
2804
HOST COMPUTER #1
2801

FIG. 28

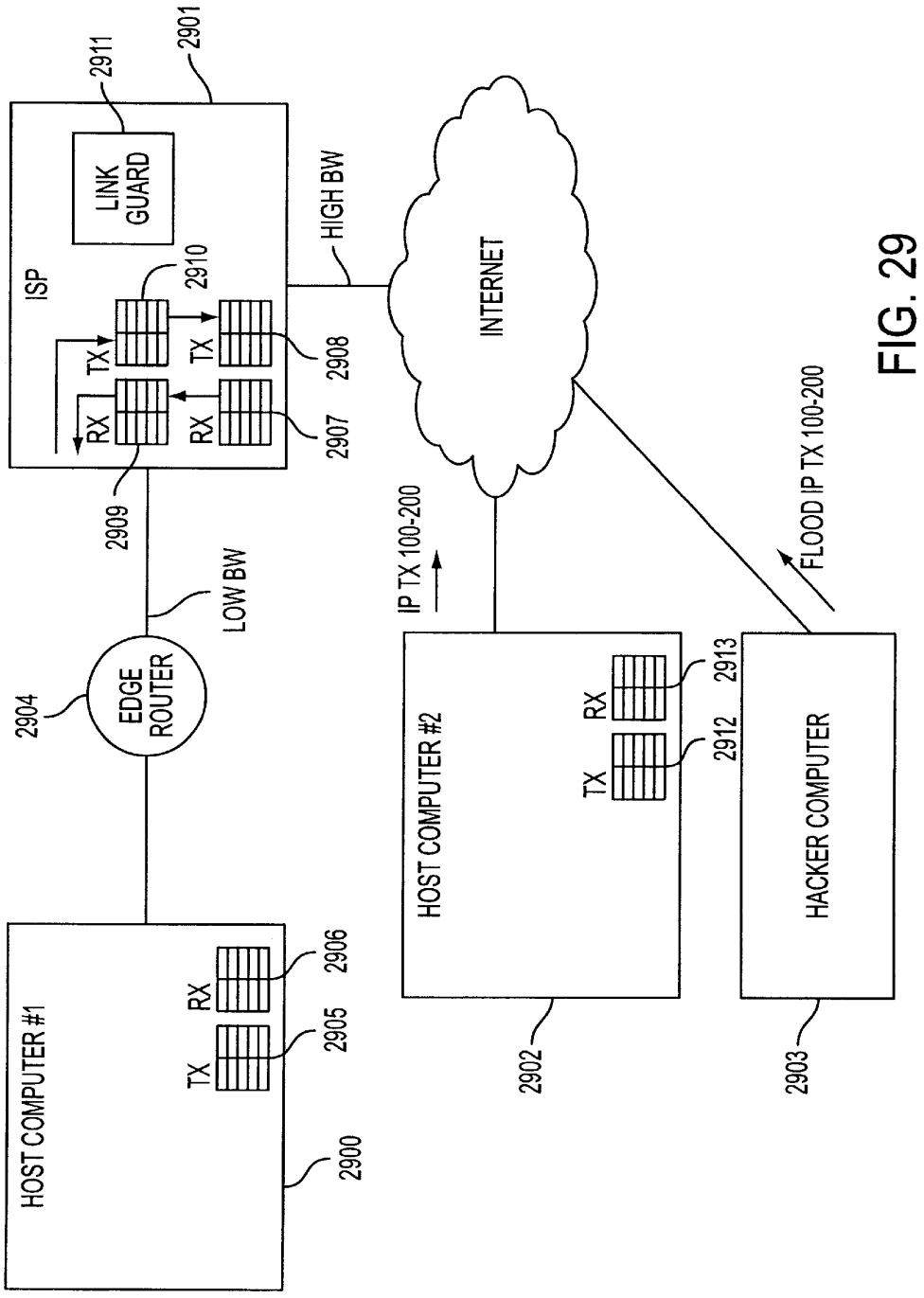

2901
2911
LINK GUARD
ISP
2910
TX
RX
TX
RX
2908
2907
HIGH BW
INTERNET
2909
LOW BW
2904
EDGE ROUTER
IP TX 100-200
FLOOD IP TX 100-200
HOST COMPUTER #1
RX
TX
2906
2905
2900
HOST COMPUTER #2
RX
TX
2913
2912
HACKER COMPUTER
2902
2903

FIG. 29

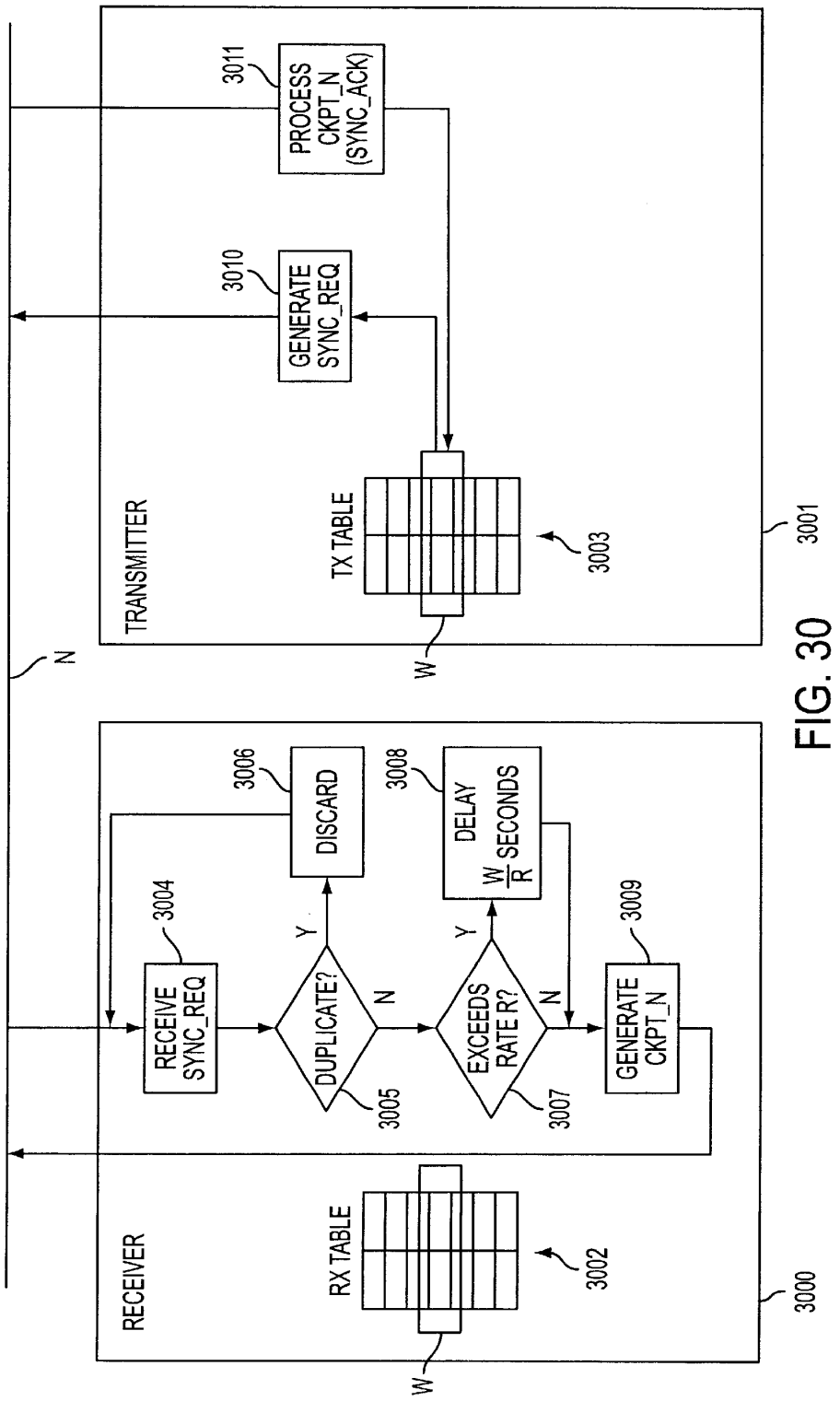

TRANSMITTER
3001
PROCESS CKPT_N (SYNC_ACK)
3011
GENERATE SYNC_REQ
3010
TX TABLE
3003
W
N
RECEIVER
3000
RX TABLE
3002
W
RECEIVE SYNC_REQ
3004
DUPLICATE?
3005
Y
N
DISCARD
3006
EXCEEDS RATE R?
3007
Y
N
DELAY W/R SECONDS
3008
GENERATE CKPT_N
3009

FIG. 30

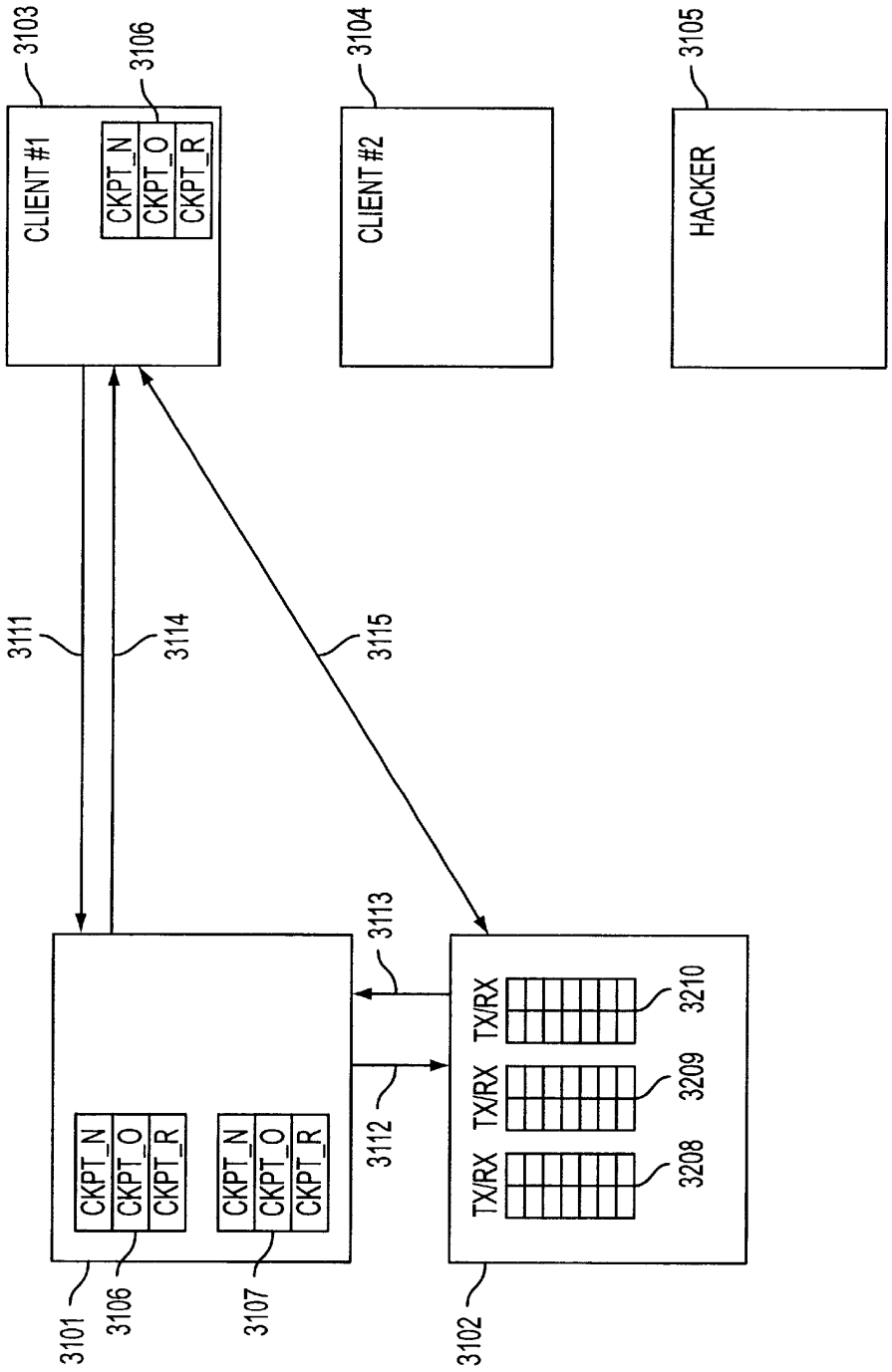

3103
3106
3104
3105
CLIENT #1
CKPT_N
CKPT_O
CKPT_R
CLIENT #2
HACKER
3111
3114
3115
3113
3112
TX/RX
TX/RX
TX/RX
3210
3209
3208
CKPT_N
CKPT_O
CKPT_R
CKPT_N
CKPT_O
CKPT_R
3101
3106
3107
3102

FIG. 31

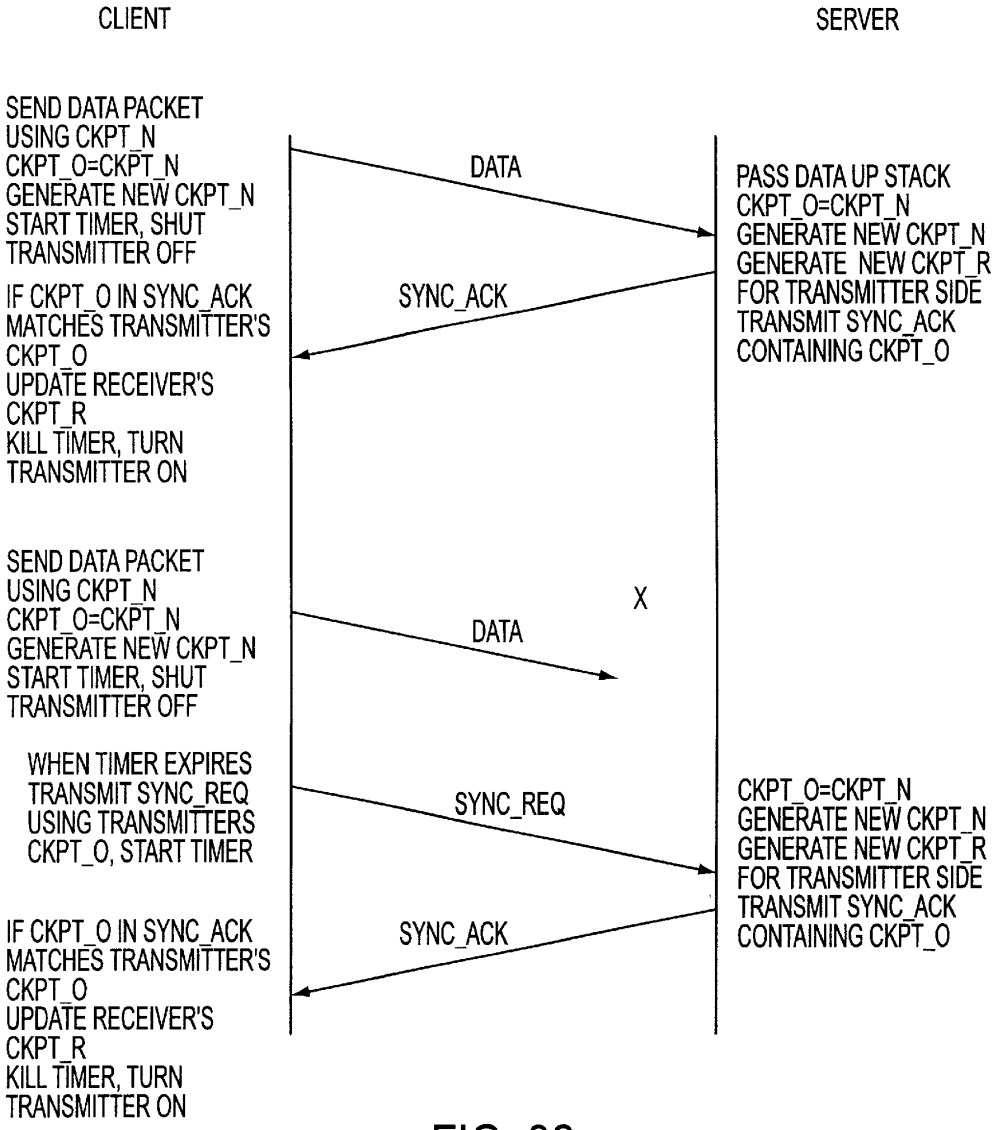

CLIENT
SERVER
SEND DATA PACKET
USING CKPT_N
CKPT_O=CKPT_N
GENERATE NEW CKPT_N
START TIMER, SHUT
TRANSMITTER OFF
DATA
PASS DATA UP STACK
CKPT_O=CKPT_N
GENERATE NEW CKPT_N
GENERATE NEW CKPT_R
FOR TRANSMITTER SIDE
TRANSMIT SYNC_ACK
CONTAINING CKPT_O
IF CKPT_O IN SYNC_ACK
MATCHES TRANSMITTER'S
CKPT_O
UPDATE RECEIVER'S
CKPT_R
KILL TIMER, TURN
TRANSMITTER ON
SYNC_ACK
SEND DATA PACKET
USING CKPT_N
CKPT_O=CKPT_N
GENERATE NEW CKPT_N
START TIMER, SHUT
TRANSMITTER OFF
X
DATA
WHEN TIMER EXPIRES
TRANSMIT SYNC_REQ
USING TRANSMITTERS
CKPT_O, START TIMER
SYNC_REQ
CKPT_O=CKPT_N
GENERATE NEW CKPT_N
GENERATE NEW CKPT_R
FOR TRANSMITTER SIDE
TRANSMIT SYNC_ACK
CONTAINING CKPT_O
IF CKPT_O IN SYNC_ACK
MATCHES TRANSMITTER'S
CKPT_O
UPDATE RECEIVER'S
CKPT_R
KILL TIMER, TURN
TRANSMITTER ON
SYNC_ACK

FIG. 32

AGILE NETWORK PROTOCOL FOR SECURE COMMUNICATIONS WITH ASSURED SYSTEM AVAILABILITY

CROSS-REFERENCE TO RELATED APPLICATION

This application claims priority from and is a continuation-in-part of previously filed U.S. application Ser. No. 09/429,643, filed on Oct. 29, 1999. The subject matter of that application, which is bodily incorporated herein, derives from provisional U.S. application No. 60/106,261 (filed Oct. 30, 1998) and No. 60/137,704 (filed Jun. 7, 1999).

BACKGROUND OF THE INVENTION

A tremendous variety of methods have been proposed and implemented to provide security and anonymity for communications over the Internet. The variety stems, in part, from the different needs of different Internet users. A basic heuristic framework to aid in discussing these different security techniques is illustrated in FIG. **1**. Two terminals, an originating terminal **100** and a destination terminal **110** are in communication over the Internet. It is desired for the communications to be secure, that is, immune to eavesdropping. For example, terminal **100** may transmit secret information to terminal **110** over the Internet **107**. Also, it may be desired to prevent an eavesdropper from discovering that terminal **100** is in communication with terminal **110**. For example, if terminal **100** is a user and terminal **110** hosts a web site, terminal **100**'s user may not want anyone in the intervening networks to know what web sites he is "visiting." Anonymity would thus be an issue, for example, for companies that want to keep their market research interests private and thus would prefer to prevent outsiders from knowing which web-sites or other Internet resources they are "visiting." These two security issues may be called data security and anonymity, respectively.

Data security is usually tackled using some form of data encryption. An encryption key **48** is known at both the originating and terminating terminals **100** and **110**. The keys may be private and public at the originating and destination terminals **100** and **110**, respectively or they may be symmetrical keys (the same key is used by both parties to encrypt and decrypt). Many encryption methods are known and usable in this context.

To hide traffic from a local administrator or ISP, a user can employ a local proxy server in communicating over an encrypted channel with an outside proxy such that the local administrator or ISP only sees the encrypted traffic. Proxy servers prevent destination servers from determining the identities of the originating clients. This system employs an intermediate server interposed between client and destination server. The destination server sees only the Internet Protocol (IP) address of the proxy server and not the originating client. The target server only sees the address of the outside proxy. This scheme relies on a trusted outside proxy server. Also, proxy schemes are vulnerable to traffic analysis methods of determining identities of transmitters and receivers. Another important limitation of proxy servers is that the server knows the identities of both calling and called parties. In many instances, an originating terminal, such as terminal A, would prefer to keep its identity concealed from the proxy, for example, if the proxy server is provided by an Internet service provider (ISP).

To defeat traffic analysis, a scheme called Chaum's mixes employs a proxy server that transmits and receives fixed length messages, including dummy messages. Multiple originating terminals are connected through a mix (a server) to multiple target servers. It is difficult to tell which of the originating terminals are communicating to which of the connected target servers, and the dummy messages confuse eavesdroppers' efforts to detect communicating pairs by analyzing traffic. A drawback is that there is a risk that the mix server could be compromised. One way to deal with this risk is to spread the trust among multiple mixes. If one mix is compromised, the identities of the originating and target terminals may remain concealed. This strategy requires a number of alternative mixes so that the intermediate servers interposed between the originating and target terminals are not determinable except by compromising more than one mix. The strategy wraps the message with multiple layers of encrypted addresses. The first mix in a sequence can decrypt only the outer layer of the message to reveal the next destination mix in sequence. The second mix can decrypt the message to reveal the next mix and so on. The target server receives the message and, optionally, a multi-layer encrypted payload containing return information to send data back in the same fashion. The only way to defeat such a mix scheme is to collude among mixes. If the packets are all fixed-length and intermixed with dummy packets, there is no way to do any kind of traffic analysis.

Still another anonymity technique, called 'crowds,' protects the identity of the originating terminal from the intermediate proxies by providing that originating terminals belong to groups of proxies called crowds. The crowd proxies are interposed between originating and target terminals. Each proxy through which the message is sent is randomly chosen by an upstream proxy. Each intermediate proxy can send the message either to another randomly chosen proxy in the "crowd" or to the destination. Thus, even crowd members cannot determine if a preceding proxy is the originator of the message or if it was simply passed from another proxy.

ZKS (Zero-Knowledge Systems) Anonymous IP Protocol allows users to select up to any of five different pseudonyms, while desktop software encrypts outgoing traffic and wraps it in User Datagram Protocol (UDP) packets. The first server in a 2+-hop system gets the UDP packets, strips off one layer of encryption to add another, then sends the traffic to the next server, which strips off yet another layer of encryption and adds a new one. The user is permitted to control the number of hops. At the final server, traffic is decrypted with an untraceable IP address. The technique is called onion-routing. This method can be defeated using traffic analysis. For a simple example, bursts of packets from a user during low-duty periods can reveal the identities of sender and receiver.

Firewalls attempt to protect LANs from unauthorized access and hostile exploitation or damage to computers connected to the LAN. Firewalls provide a server through which all access to the LAN must pass. Firewalls are centralized systems that require administrative overhead to maintain. They can be compromised by virtual-machine applications ("applets"). They instill a false sense of security that leads to security breaches for example by users sending sensitive information to servers outside the firewall or encouraging use of modems to sidestep the firewall security. Firewalls are not useful for distributed systems such as business travelers, extranets, small teams, etc.

SUMMARY OF THE INVENTION

A secure mechanism for communicating over the internet, including a protocol referred to as the Tunneled Agile

Routing Protocol (TARP), uses a unique two-layer encryption format and special TARP routers. TARP routers are similar in function to regular IP routers. Each TARP router has one or more IP addresses and uses normal IP protocol to send IP packet messages ("packets" or "datagrams"). The IP packets exchanged between TARP terminals via TARP routers are actually encrypted packets whose true destination address is concealed except to TARP routers and servers. The normal or "clear" or "outside" IP header attached to TARP IP packets contains only the address of a next hop router or destination server. That is, instead of indicating a final destination in the destination field of the IP header, the TARP packet's IP header always points to a next-hop in a series of TARP router hops, or to the final destination. This means there is no overt indication from an intercepted TARP packet of the true destination of the TARP packet since the destination could always be next-hop TARP router as well as the final destination.

Each TARP packet's true destination is concealed behind a layer of encryption generated using a link key. The link key is the encryption key used for encrypted communication between the hops intervening between an originating TARP terminal and a destination TARP terminal. Each TARP router can remove the outer layer of encryption to reveal the destination router for each TARP packet. To identify the link key needed to decrypt the outer layer of encryption of a TARP packet, a receiving TARP or routing terminal may identify the transmitting terminal by the sender/receiver IP numbers in the cleartext IP header.

Once the outer layer of encryption is removed, the TARP router determines the final destination. Each TARP packet **140** undergoes a minimum number of hops to help foil traffic analysis. The hops may be chosen at random or by a fixed value. As a result, each TARP packet may make random trips among a number of geographically disparate routers before reaching its destination. Each trip is highly likely to be different for each packet composing a given message because each trip is independently randomly determined. This feature is called agile routing. The fact that different packets take different routes provides distinct advantages by making it difficult for an interloper to obtain all the packets forming an entire multi-packet message. The associated advantages have to do with the inner layer of encryption discussed below. Agile routing is combined with another feature that furthers this purpose; a feature that ensures that any message is broken into multiple packets.

The IP address of a TARP router can be changed, a feature called IP agility. Each TARP router, independently or under direction from another TARP terminal or router, can change its IP address. A separate, unchangeable identifier or address is also defined. This address, called the TARP address, is known only to TARP routers and terminals and may be correlated at any time by a TARP router or a TARP terminal using a Lookup Table (LUT). When a TARP router or terminal changes its IP address, it updates the other TARP routers and terminals which in turn update their respective LUTs.

The message payload is hidden behind an inner layer of encryption in the TARP packet that can only be unlocked using a session key. The session key is not available to any of the intervening TARP routers. The session key is used to decrypt the payloads of the TARP packets permitting the data stream to be reconstructed.

Communication may be made private using link and session keys, which in turn may be shared and used according to any desired method. For example, public/private keys or symmetric keys may be used.

To transmit a data stream, a TARP originating terminal constructs a series of TARP packets from a series of IP packets generated by a network (IP) layer process. (Note that the terms "network layer," "data link layer," "application layer," etc. used in this specification correspond to the Open Systems Intercomection (OSI) network terminology.) The payloads of these packets are assembled into a block and chain-block encrypted using the session key. This assumes, of course, that all the IP packets are destined for the same TARP terminal. The block is then interleaved and the interleaved encrypted block is broken into a series of payloads, one for each TARP packet to be generated. Special TARP headers IPT are then added to each payload using the IP headers from the data stream packets. The TARP headers can be identical to normal IP headers or customized in some way. They should contain a formula or data for deinterleaving the data at the destination TARP terminal, a time-to-live (TTL) parameter to indicate the number of hops still to be executed, a data type identifier which indicates whether the payload contains, for example, TCP or UDP data, the sender's TARP address, the destination TARP address, and an indicator as to whether the packet contains real or decoy data or a formula for filtering out decoy data if decoy data is spread in some way through the TARP payload data.

Note that although chain-block encryption is discussed here with reference to the session key, any encryption method may be used. Preferably, as in chain block encryption, a method should be used that makes unauthorized decryption difficult without an entire result of the encryption process. Thus, by separating the encrypted block among multiple packets and making it difficult for an interloper to obtain access to all of such packets, the contents of the communications are provided an extra layer of security.

Decoy or dummy data can be added to a stream to help foil traffic analysis by reducing the peak-to-average network load. It may be desirable to provide the TARP process with an ability to respond to the time of day or other criteria to generate more decoy data during low traffic periods so that communication bursts at one point in the Internet cannot be tied to communication bursts at another point to reveal the communicating endpoints.

Dummy data also helps to break the data into a larger number of inconspicuously-sized packets permitting the interleave window size to be increased while maintaining a reasonable size for each packet. (The packet size can be a single standard size or selected from a fixed range of sizes.) One primary reason for desiring for each message to be broken into multiple packets is apparent if a chain block encryption scheme is used to form the first encryption layer prior to interleaving. A single block encryption may be applied to portion, or entirety, of a message, and that portion or entirety then interleaved into a number of separate packets. Considering the agile IP routing of the packets, and the attendant difficulty of reconstructing an entire sequence of packets to form a single block-encrypted message element, decoy packets can significantly increase the difficulty of reconstructing an entire data stream.

The above scheme may be implemented entirely by processes operating between the data link layer and the network layer of each server or terminal participating in the TARP system. Because the encryption system described above is insertable between the data link and network layers, the processes involved in supporting the encrypted communication may be completely transparent to processes at the IP (network) layer and above. The TARP processes may also be completely transparent to the data link layer processes as

well. Thus, no operations at or above the Network layer, or at or below the data link layer, are affected by the insertion of the TARP stack. This provides additional security to all processes at or above the network layer, since the difficulty of unauthorized penetration of the network layer (by, for example, a hacker) is increased substantially. Even newly developed servers running at the session layer leave all processes below the session layer vulnerable to attack. Note that in this architecture, security is distributed. That is, notebook computers used by executives on the road, for example, can communicate over the Internet without any compromise in security.

IP address changes made by TARP terminals and routers can be done at regular intervals, at random intervals, or upon detection of "attacks." The variation of IP addresses hinders traffic analysis that might reveal which computers are communicating, and also provides a degree of immunity from attack. The level of immunity from attack is roughly proportional to the rate at which the IP address of the host is changing.

As mentioned, IP addresses may be changed in response to attacks. An attack may be revealed, for example, by a regular series of messages indicating that a router is being probed in some way. Upon detection of an attack, the TARP layer process may respond to this event by changing its IP address. In addition, it may create a subprocess that maintains the original IP address and continues interacting with the attacker in some manner.

Decoy packets may be generated by each TARP terminal on some basis determined by an algorithm. For example, the algorithm may be a random one which calls for the generation of a packet on a random basis when the terminal is idle. Alternatively, the algorithm may be responsive to time of day or detection of low traffic to generate more decoy packets during low traffic times. Note that packets are preferably generated in groups, rather than one by one, the groups being sized to simulate real messages. In addition, so that decoy packets may be inserted in normal TARP message streams, the background loop may have a latch that makes it more likely to insert decoy packets when a message stream is being received. Alternatively, if a large number of decoy packets is received along with regular TARP packets, the algorithm may increase the rate of dropping of decoy packets rather than forwarding them. The result of dropping and generating decoy packets in this way is to make the apparent incoming message size different from the apparent outgoing message size to help foil traffic analysis.

In various other embodiments of the invention, a scalable version of the system may be constructed in which a plurality of IP addresses are preassigned to each pair of communicating nodes in the network. Each pair of nodes agrees upon an algorithm for "hopping" between IP addresses (both sending and receiving), such that an eavesdropper sees apparently continuously random IP address pairs (source and destination) for packets transmitted between the pair. Overlapping or "reusable" IP addresses may be allocated to different users on the same subnet, since each node merely verifies that a particular packet includes a valid source/destination pair from the agreed-upon algorithm. Source/destination pairs are preferably not reused between any two nodes during any given end-to-end session, though limited IP block sizes or lengthy sessions might require it.

Further improvements described in this continuation-in-part application include: (1) a load balancer that distributes packets across different transmission paths according to transmission path quality; (2) a DNS proxy server that transparently creates a virtual private network in response to a domain name inquiry; (3) a large-to-small link bandwidth management feature that prevents denial-of-service attacks at system chokepoints; (4) a traffic limiter that regulates incoming packets by limiting the rate at which a transmitter can be synchronized with a receiver; and (5) a signaling synchronizer that allows a large number of nodes to communicate with a central node by partitioning the communication function between two separate entities

BRIEF DESCRIPTION OF THE DRAWINGS

FIG. **1** is an illustration of secure communications over the Internet according to a prior art embodiment.

FIG. **2** is an illustration of secure communications over the Internet according to a an embodiment of the invention.

FIG. **3***a* is an illustration of a process of forming a tunneled IP packet according to an embodiment of the invention.

FIG. **3***b* is an illustration of a process of forming a tunneled IP packet according to another embodiment of the invention.

FIG. **4** is an illustration of an OSI layer location of processes that may be used to implement the invention.

FIG. **5** is a flow chart illustrating a process for routing a tunneled packet according to an embodiment of the invention.

FIG. **6** is a flow chart illustrating a process for forming a tunneled packet according to an embodiment of the invention.

FIG. **7** is a flow chart illustrating a process for receiving a tunneled packet according to an embodiment of the invention.

FIG. **8** shows how a secure session is established and synchronized between a client and a TARP router.

FIG. **9** shows an IP address hopping scheme between a client computer and TARP router using transmit and receive tables in each computer.

FIG. **10** shows physical link redundancy among three Internet Service Providers (ISPs) and a client computer.

FIG. **11** shows how multiple IP packets can be embedded into a single "frame" such as an Ethernet frame, and further shows the use of a discriminator field to camouflage true packet recipients.

FIG. **12**A shows a system that employs hopped hardware addresses, hopped IP addresses, and hopped discriminator fields.

FIG. **12**B shows several different approaches for hopping hardware addresses, IP addresses, and discriminator fields in combination.

FIG. **13** shows a technique for automatically re-establishing synchronization between sender and receiver through the use of a partially public sync value.

FIG. **14** shows a "checkpoint" scheme for regaining synchronization between a sender and recipient.

FIG. **15** shows further details of the checkpoint scheme of FIG. **14**.

FIG. **16** shows how two addresses can be decomposed into a plurality of segments for comparison with presence vectors.

FIG. **17** shows a storage array for a receiver's active addresses.

FIG. **18** shows the receiver's storage array after receiving a sync request.

FIG. **19** shows the receiver's storage array after new addresses have been generated.

FIG. **20** shows a system employing distributed transmission paths.

FIG. **21** shows a plurality of link transmission tables that can be used to route packets in the system of FIG. **20**.

FIG. **22**A shows a flowchart for adjusting weight value distributions associated with a plurality of transmission links.

FIG. **22**B shows a flowchart for setting a weight value to zero if a transmitter turns off.

FIG. **23** shows a system employing distributed transmission paths with adjusted weight value distributions for each path.

FIG. **24** shows an example using the system of FIG. **23**.

FIG. **25** shows a conventional domain-name look-up service.

FIG. **26** shows a system employing a DNS proxy server with transparent VPN creation.

FIG. **27** shows steps that can be carried out to implement transparent VPN creation based on a DNS look-up function.

FIG. **28** shows a system including a link guard function that prevents packet overloading on a low-bandwidth link LOW BW.

FIG. **29** shows one embodiment of a system employing the principles of FIG. **28**.

FIG. **30** shows a system that regulates packet transmission rates by throttling the rate at which synchronizations are performed.

FIG. **31** shows a signaling server **3101** and a transport server **3102** used to establish a VPN with a client computer.

FIG. **32** shows message flows relating to synchronization protocols of FIG. **31**.

DETAILED DESCRIPTION OF THE INVENTION

Referring to FIG. **2**, a secure mechanism for communicating over the internet employs a number of special routers or servers, called TARP routers **122–127** that are similar to regular IP routers **128–132** in that each has one or more IP addresses and uses normal IP protocol to send normal-looking IP packet messages, called TARP packets **140**. TARP packets **140** are identical to normal IP packet messages that are routed by regular IP routers **128–132** because each TARP packet **140** contains a destination address as in a normal IP packet. However, instead of indicating a final destination in the destination field of the IP header, the TARP packet's **140** IP header always points to a next-hop in a series of TARP router hops, or the final destination, TARP terminal **110**. Because the header of the TARP packet contains only the next-hop destination, there is no overt indication from an intercepted TARP packet of the true destination of the TARP packet **140** since the destination could always be the next-hop TARP router as well as the final destination, TARP terminal **110**.

Each TARP packet's true destination is concealed behind an outer layer of encryption generated using a link key **146**. The link key **146** is the encryption key used for encrypted communication between the end points (TARP terminals or TARP routers) of a single link in the chain of hops connecting the originating TARP terminal **100** and the destination TARP terminal **110**. Each TARP router **122–127**, using the link key **146** it uses to communicate with the previous hop in a chain, can use the link key to reveal the true destination of a TARP packet. To identify the link key needed to decrypt the outer layer of encryption of a TARP packet, a receiving TARP or routing terminal may identify the transmitting terminal (which may indicate the link key used) by the sender field of the clear IP header. Alternatively, this identity may be hidden behind another layer of encryption in available bits in the clear IP header. Each TARP router, upon receiving a TARP message, determines if the message is a TARP message by using authentication data in the TARP packet. This could be recorded in available bytes in the TARP packet's IP header. Alternatively, TARP packets could be authenticated by attempting to decrypt using the link key **146** and determining if the results are as expected. The former may have computational advantages because it does not involve a decryption process.

Once the outer layer of decryption is completed by a TARP router **122–127**, the TARP router determines the final destination. The system is preferably designed to cause each TARP packet **140** to undergo a minimum number of hops to help foil traffic analysis. The time to live counter in the IP header of the TARP message may be used to indicate a number of TARP router hops yet to be completed. Each TARP router then would decrement the counter and determine from that whether it should forward the TARP packet **140** to another TARP router **122–127** or to the destination TARP terminal **110**. If the time to live counter is zero or below zero after decrementing, for an example of usage, the TARP router receiving the TARP packet **140** may forward the TARP packet **140** to the destination TARP terminal **110**. If the time to live counter is above zero after decrementing, for an example of usage, the TARP router receiving the TARP packet **140** may forward the TARP packet **140** to a TARP router **122–127** that the current TARP terminal chooses at random. As a result, each TARP packet **140** is routed through some minimum number of hops of TARP routers **122–127** which are chosen at random.

Thus, each TARP packet, irrespective of the traditional factors determining traffic in the Internet, makes random trips among a number of geographically disparate routers before reaching its destination and each trip is highly likely to be different for each packet composing a given message because each trip is independently randomly determined as described above. This feature is called agile routing. For reasons that will become clear shortly, the fact that different packets take different routes provides distinct advantages by making it difficult for an interloper to obtain all the packets forming an entire multi-packet message. Agile routing is combined with another feature that furthers this purpose, a feature that ensures that any message is broken into multiple packets.

A TARP router receives a TARP packet when an IP address used by the TARP router coincides with the IP address in the TARP packet's IP header IP_C. The IP address of a TARP router, however, may not remain constant. To avoid and manage attacks, each TARP router, independently or under direction from another TARP terminal or router, may change its IP address. A separate, unchangeable identifier or address is also defined. This address, called the TARP address, is known only to TARP routers and terminals and may be correlated at any time by a TARP router or a TARP terminal using a Lookup Table (LUT). When a TARP router or terminal changes its IP address, it updates the other TARP routers and terminals which in turn update their respective LUTs. In reality, whenever a TARP router looks up the address of a destination in the encrypted header, it must convert a TARP address to a real IP address using its LUT.

While every TARP router receiving a TARP packet has the ability to determine the packet's final destination, the message payload is embedded behind an inner layer of encryption in the TARP packet that can only be unlocked using a session key. The session key is not available to any of the TARP routers **122**–**127** intervening between the originating **100** and destination **110** TARP terminals. The session key is used to decrypt the payloads of the TARP packets **140** permitting an entire message to be reconstructed.

In one embodiment, communication may be made private using link and session keys, which in turn may be shared and used according any desired method. For example, a public key or symmetric keys may be communicated between link or session endpoints using a public key method. Any of a variety of other mechanisms for securing data to ensure that only authorized computers can have access to the private information in the TARP packets **140** may be used as desired.

Referring to FIG. **3***a*, to construct a series of TARP packets, a data stream **300** of IP packets **207***a*, **207***b*, **207***c*, etc., such series of packets being formed by a network (IP) layer process, is broken into a series of small sized segments. In the present example, equal-sized segments **1**–**9** are defined and used to construct a set of interleaved data packets A, B, and C. Here it is assumed that the number of interleaved packets A, B, and C formed is three and that the number of IP packets **207***a*–**207***c* used to form the three interleaved packets A, B, and C is exactly three. Of course, the number of IP packets spread over a group of interleaved packets may be any convenient number as may be the number of interleaved packets over which the incoming data stream is spread. The latter, the number of interleaved packets over which the data stream is spread, is called the interleave window.

To create a packet, the transmitting software interleaves the normal IP packets **207***a* et. seq. to form a new set of interleaved payload data **320**. This payload data **320** is then encrypted using a session key to form a set of session-key-encrypted payload data **330**, each of which, A, B, and C, will form the payload of a TARP packet. Using the IP header data, from the original packets **207***a*–**207***c*, new TARP headers IP_T are formed. The TARP headers IP_T can be identical to normal IP headers or customized in some way. In a preferred embodiment, the TARP headers IP_T are IP headers with added data providing the following information required for routing and reconstruction of messages, some of which data is ordinarily, or capable of being, contained in normal IP headers:

1. A window sequence number—an identifier that indicates where the packet belongs in the original message sequence.
2. An interleave sequence number—an identifier that indicates the interleaving sequence used to form the packet so that the packet can be deinterleaved along with other packets in the interleave window.
3. A time-to-live (TTL) datum—indicates the number of TARP-router-hops to be executed before the packet reaches its destination. Note that the TTL parameter may provide a datum to be used in a probabilistic formula for determining whether to route the packet to the destination or to another hop.
4. Data type identifier—indicates whether the payload contains, for example, TCP or UDP data.
5. Sender's address—indicates the sender's address in the TARP network.
6. Destination address—indicates the destination terminal's address in the TARP network.
7. Decoy/Real—an indicator of whether the packet contains real message data or dummy decoy data or a combination.

Obviously, the packets going into a single interleave window must include only packets with a common destination. Thus, it is assumed in the depicted example that the IP headers of IP packets **207***a*–**207***c* all contain the same destination address or at least will be received by the same terminal so that they can be deinterleaved. Note that dummy or decoy data or packets can be added to form a larger interleave window than would otherwise be required by the size of a given message. Decoy or dummy data can be added to a stream to help foil traffic analysis by leveling the load on the network. Thus, it may be desirable to provide the TARP process with an ability to respond to the time of day or other criteria to generate more decoy data during low traffic periods so that communication bursts at one point in the Internet cannot be tied to communication bursts at another point to reveal the communicating endpoints.

Dummy data also helps to break the data into a larger number of inconspicuously-sized packets permitting the interleave window size to be increased while maintaining a reasonable size for each packet. (The packet size can be a single standard size or selected from a fixed range of sizes.) One primary reason for desiring for each message to be broken into multiple packets is apparent if a chain block encryption scheme is used to form the first encryption layer prior to interleaving. A single block encryption may be applied to a portion, or the entirety, of a message, and that portion or entirety then interleaved into a number of separate packets.

Referring to FIG. **3***b*, in an alternative mode of TARP packet construction, a series of IP packets are accumulated to make up a predefined interleave window. The payloads of the packets are used to construct a single block **520** for chain block encryption using the session key. The payloads used to form the block are presumed to be destined for the same terminal. The block size may coincide with the interleave window as depicted in the example embodiment of FIG. **3***b*. After encryption, the encrypted block is broken into separate payloads and segments which are interleaved as in the embodiment of FIG. **3***a*. The resulting interleaved packets A, B, and C, are then packaged as TARP packets with TARP headers as in the Example of FIG. **3***a*. The remaining process is as shown in, and discussed with reference to, FIG. **3***a*.

Once the TARP packets **340** are formed, each entire TARP packet **340**, including the TARP header IP_T, is encrypted using the link key for communication with the first-hop-TARP router. The first hop TARP router is randomly chosen. A final unencrypted IP header IP_C is added to each encrypted TARP packet **340** to form a normal IP packet **360** that can be transmitted to a TARP router. Note that the process of constructing the TARP packet **360** does not have to be done in stages as described. The above description is just a useful heuristic for describing the final product, namely, the TARP packet.

Note that, TARP header IP_T could be a completely custom header configuration with no similarity to a normal IP header except that it contain the information identified above. This is so since this header is interpreted by only TARP routers.

The above scheme may be implemented entirely by processes operating between the data link layer and the network layer of each server or terminal participating in the TARP system. Referring to FIG. **4**, a TARP transceiver **405** can be an originating terminal **100**, a destination terminal

110, or a TARP router 122–127. In each TARP Transceiver 405, a transmitting process is generated to receive normal packets from the Network (IP) layer and generate TARP packets for communication over the network. A receiving process is generated to receive normal IP packets containing TARP packets and generate from these normal IP packets which are "passed up" to the Network (IP) layer. Note that where the TARP Transceiver 405 is a router, the received TARP packets 140 are not processed into a stream of IP packets 415 because they need only be authenticated as proper TARP packets and then passed to another TARP router or a TARP destination terminal 110. The intervening process, a "TARP Layer" 420, could be combined with either the data link layer 430 or the Network layer 410. In either case, it would intervene between the data link layer 430 so that the process would receive regular IP packets containing embedded TARP packets and "hand up" a series of reassembled IP packets to the Network layer 410. As an example of combining the TARP layer 420 with the data link layer 430, a program may augment the normal processes running a communications card, for example, an Ethernet card. Alternatively, the TARP layer processes may form part of a dynamically loadable module that is loaded and executed to support communications between the network and data link layers.

Because the encryption system described above can be inserted between the data link and network layers, the processes involved in supporting the encrypted communication may be completely transparent to processes at the IP (network) layer and above. The TARP processes may also be completely transparent to the data link layer processes as well. Thus, no operations at or above the network layer, or at or below the data link layer, are affected by the insertion of the TARP stack. This provides additional security to all processes at or above the network layer, since the difficulty of unauthorized penetration of the network layer (by, for example, a hacker) is increased substantially. Even newly developed servers running at the session layer leave all processes below the session layer vulnerable to attack. Note that in this architecture, security is distributed. That is, notebook computers used by executives on the road, for example, can communicate over the Internet without any compromise in security.

Note that IP address changes made by TARP terminals and routers can be done at regular intervals, at random intervals, or upon detection of "attacks." The variation of IP addresses hinders traffic analysis that might reveal which computers are communicating, and also provides a degree of immunity from attack. The level of immunity from attack is roughly proportional to the rate at which the IP address of the host is changing.

As mentioned, IP addresses may be changed in response to attacks. An attack may be revealed, for example, by a regular series of messages indicates that a router is being probed in some way. Upon detection of an attack, the TARP layer process may respond to this event by changing its IP address. To accomplish this, the TARP process will construct a TARP-formatted message, in the style of Internet Control Message Protocol (ICMP) datagrams as an example; this message will contain the machine's TARP address, its previous IP address, and its new IP address. The TARP layer will transmit this packet to at least one known TARP router; then upon receipt and validation of the message, the TARP router will update its LUT with the new IP address for the stated TARP address. The TARP router will then format a similar message, and broadcast it to the other TARP routers so that they may update their LUTs. Since the total number of TARP routers on any given subnet is expected to be relatively small, this process of updating the LUTs should be relatively fast. It may not, however, work as well when there is a relatively large number of TARP routers and/or a relatively large number of clients; this has motivated a refinement of this architecture to provide scalability; this refinement has led to a second embodiment, which is discussed below.

Upon detection of an attack, the TARP process may also create a subprocess that maintains the original IP address and continues interacting with the attacker. The latter may provide an opportunity to trace the attacker or study the attacker's methods (called "fishbowling" drawing upon the analogy of a small fish in a fish bowl that "thinks" it is in the ocean but is actually under captive observation). A history of the communication between the attacker and the abandoned (fishbowled) IP address can be recorded or transmitted for human analysis or further synthesized for purposes of responding in some way.

As mentioned above, decoy or dummy data or packets can be added to outgoing data streams by TARP terminals or routers. In addition to making it convenient to spread data over a larger number of separate packets, such decoy packets can also help to level the load on inactive portions of the Internet to help foil traffic analysis efforts.

Decoy packets may be generated by each TARP terminal 100, 110 or each router 122–127 on some basis determined by an algorithm. For example, the algorithm may be a random one which calls for the generation of a packet on a random basis when the terminal is idle. Alternatively, the algorithm may be responsive to time of day or detection of low traffic to generate more decoy packets during low traffic times. Note that packets are preferably generated in groups, rather than one by one, the groups being sized to simulate real messages. In addition, so that decoy packets may be inserted in normal TARP message streams, the background loop may have a latch that makes it more likely to insert decoy packets when a message stream is being received. That is, when a series of messages are received, the decoy packet generation rate may be increased. Alternatively, if a large number of decoy packets is received along with regular TARP packets, the algorithm may increase the rate of dropping of decoy packets rather than forwarding them. The result of dropping and generating decoy packets in this way is to make the apparent incoming message size different from the apparent outgoing message size to help foil traffic analysis. The rate of reception of packets, decoy or otherwise, may be indicated to the decoy packet dropping and generating processes through perishable decoy and regular packet counters. (A perishable counter is one that resets or decrements its value in response to time so that it contains a high value when it is incremented in rapid succession and a small value when incremented either slowly or a small number of times in rapid succession.) Note that destination TARP terminal 110 may generate decoy packets equal in number and size to those TARP packets received to make it appear it is merely routing packets and is therefore not the destination terminal.

Referring to FIG. 5, the following particular steps may be employed in the above-described method for routing TARP packets.

S0. A background loop operation is performed which applies an algorithm which determines the generation of decoy IP packets. The loop is interrupted when an encrypted TARP packet is received.

S2. The TARP packet may be probed in some way to authenticate the packet before attempting to decrypt it

using the link key. That is, the router may determine that the packet is an authentic TARP packet by performing a selected operation on some data included with the clear IP header attached to the encrypted TARP packet contained in the payload. This makes it possible to avoid performing decryption on packets that are not authentic TARP packets.

S3. The TARP packet is decrypted to expose the destination TARP address and an indication of whether the packet is a decoy packet or part of a real message.

S4. If the packet is a decoy packet, the perishable decoy counter is incremented.

S5. Based on the decoy generation/dropping algorithm and the perishable decoy counter value, if the packet is a decoy packet, the router may choose to throw it away. If the received packet is a decoy packet and it is determined that it should be thrown away (S6), control returns to step S0.

S7. The TTL parameter of the TARP header is decremented and it is determined if the TTL parameter is greater than zero.

S8. If the TTL parameter is greater than zero, a TARP address is randomly chosen from a list of TARP addresses maintained by the router and the link key and IP address corresponding to that TARP address memorized for use in creating a new IP packet containing the TARP packet.

S9. If the TTL parameter is zero or less, the link key and IP address corresponding to the TARP address of the destination are memorized for use in creating the new IP packet containing the TARP packet.

S10. The TARP packet is encrypted using the memorized link key.

S11. An IP header is added to the packet that contains the stored IP address, the encrypted TARP packet wrapped with an IP header, and the completed packet transmitted to the next hop or destination.

Referring to FIG. **6**, the following particular steps may be employed in the above-described method for generating TARP packets.

S20. A background loop operation applies an algorithm that determines the generation of decoy IP packets. The loop is interrupted when a data stream containing IP packets is received for transmission.

S21. The received IP packets are grouped into a set consisting of messages with a constant IP destination address. The set is further broken down to coincide with a maximum size of an interleave window The set is encrypted, and interleaved into a set of payloads destined to become TARP packets.

S22. The TARP address corresponding to the IP address is determined from a lookup table and stored to generate the TARP header. An initial TTL count is generated and stored in the header. The TTL count may be random with minimum and maximum values or it may be fixed or determined by some other parameter.

S23. The window sequence numbers and interleave sequence numbers are recorded in the TARP headers of each packet.

S24. One TARP router address is randomly chosen for each TARP packet and the IP address corresponding to it stored for use in the clear IP header. The link key corresponding to this router is identified and used to encrypt TARP packets containing interleaved and encrypted data and TARP headers.

S25. A clear IP header with the first hop router's real IP address is generated and added to each of the encrypted TARP packets and the resulting packets.

Referring to FIG. **7**, the following particular steps may be employed in the above-described method for receiving TARP packets.

S40. A background loop operation is performed which applies an algorithm which determines the generation of decoy IP packets. The loop is interrupted when an encrypted TARP packet is received.

S42. The TARP packet may be probed to authenticate the packet before attempting to decrypt it using the link key.

S43. The TARP packet is decrypted with the appropriate link key to expose the destination TARP address and an indication of whether the packet is a decoy packet or part of a real message.

S44. If the packet is a decoy packet, the perishable decoy counter is incremented.

S45. Based on the decoy generation/dropping algorithm and the perishable decoy counter value, if the packet is a decoy packet, the receiver may choose to throw it away.

S46. The TARP packets are cached until all packets forming an interleave window are received.

S47. Once all packets of an interleave window are received, the packets are deinterleaved.

S48. The packets block of combined packets defining the interleave window is then decrypted using the session key.

S49. The decrypted block is then divided using the window sequence data and the IP_T headers are converted into normal IP_C headers. The window sequence numbers are integrated in the IP_C headers.

S50. The packets are then handed up to the IP layer processes.

1. SCALABILITY ENHANCEMENTS

The IP agility feature described above relies on the ability to transmit IP address changes to all TARP routers. The embodiments including this feature will be referred to as "boutique" embodiments due to potential limitations in scaling these features up for a large network, such as the Internet. (The "boutique" embodiments would, however, be robust for use in smaller networks, such as small virtual private networks, for example). One problem with the boutique embodiments is that if IP address changes are to occur frequently, the message traffic required to update all routers sufficiently quickly creates a serious burden on the Internet when the TARP router and/or client population gets large. The bandwidth burden added to the networks, for example in ICMP packets, that would be used to update all the TARP routers could overwhelm the Internet for a large scale implementation that approached the scale of the Internet. In other words, the boutique system's scalability is limited.

A system can be constructed which trades some of the features of the above embodiments to provide the benefits of IP agility without the additional messaging burden. This is accomplished by IP address-hopping according to shared algorithms that govern IP addresses used between links participating in communications sessions between nodes such as TARP nodes. (Note that the IP hopping technique is also applicable to the boutique embodiment.) The IP agility feature discussed with respect to the boutique system can be

modified so that it becomes decentralized under this scalable regime and governed by the above-described shared algorithm. Other features of the boutique system may be combined with this new type of IP-agility.

The new embodiment has the advantage of providing IP agility governed by a local algorithm and set of IP addresses exchanged by each communicating pair of nodes. This local governance is session-independent in that it may govern communications between a pair of nodes, irrespective of the session or end points being transferred between the directly communicating pair of nodes.

In the scalable embodiments, blocks of IP addresses are allocated to each node in the network. (This scalability will increase in the future, when Internet Protocol addresses are increased to 128-bit fields, vastly increasing the number of distinctly addressable nodes). Each node can thus use any of the IP addresses assigned to that node to communicate with other nodes in the network. Indeed, each pair of communicating nodes can use a plurality of source IP addresses and destination IP addresses for communicating with each other.

Each communicating pair of nodes in a chain participating in any session stores two blocks of IP addresses, called netblocks, and an algorithm and randomization seed for selecting, from each netblock, the next pair of source/destination IP addresses that will be used to transmit the next message. In other words, the algorithm governs the sequential selection of IP-address pairs, one sender and one receiver IP address, from each netblock. The combination of algorithm, seed, and netblock (IP address block) will be called a "hopblock." A router issues separate transmit and receive hopblocks to its clients. The send address and the receive address of the IP header of each outgoing packet sent by the client are filled with the send and receive IP addresses generated by the algorithm. The algorithm is "clocked" (indexed) by a counter so that each time a pair is used, the algorithm turns out a new transmit pair for the next packet to be sent.

The router's receive hopblock is identical to the client's transmit hopblock. The router uses the receive hopblock to predict what the send and receive IP address pair for the next expected packet from that client will be. Since packets can be received out of order, it is not possible for the router to predict with certainty what IP address pair will be on the next sequential packet. To account for this problem, the router generates a range of predictions encompassing the number of possible transmitted packet send/receive addresses, of which the next packet received could leap ahead. Thus, if there is a vanishingly small probability that a given packet will arrive at the router ahead of 5 packets transmitted by the client before the given packet, then the router can generate a series of 6 send/receive IP address pairs (or "hop window") to compare with the next received packet. When a packet is received, it is marked in the hop window as such, so that a second packet with the same IP address pair will be discarded. If an out-of-sequence packet does not arrive within a predetermined timeout period, it can be requested for retransmission or simply discarded from the receive table, depending upon the protocol in use for that communications session, or possibly by convention.

When the router receives the client's packet, it compares the send and receive IP addresses of the packet with the next N predicted send and receive IP address pairs and rejects the packet if it is not a member of this set. Received packets that do not have the predicted source/destination IP addresses falling with the window are rejected, thus thwarting possible hackers. (With the number of possible combinations, even a fairly large window would be hard to fall into at random.) If it is a member of this set, the router accepts the packet and processes it further. This link-based IP-hopping strategy, referred to as "IHOP," is a network element that stands on its own and is not necessarily accompanied by elements of the boutique system described above. If the routing agility feature described in connection with the boutique embodiment is combined with this link-based IP-hopping strategy, the router's next step would be to decrypt the TARP header to determine the destination TARP router for the packet and determine what should be the next hop for the packet. The TARP router would then forward the packet to a random TARP router or the destination TARP router with which the source TARP router has a link-based IP hopping communication established.

FIG. **8** shows how a client computer **801** and a TARP router **811** can establish a secure session. When client **801** seeks to establish an IHOP session with TARP router **811**, the client **801** sends "secure synchronization" request ("SSYN") packet **821** to the TARP router **811**. This SYN packet **821** contains the client's **801** authentication token, and may be sent to the router **811** in an encrypted format. The source and destination IP numbers on the packet **821** are the client's **801** current fixed IP address, and a "known" fixed IP address for the router **811**. (For security purposes, it may be desirable to reject any packets from outside of the local network that are destined for the router's known fixed IP address.) Upon receipt and validation of the client's **801** SSYN packet **821**, the router **811** responds by sending an encrypted "secure synchronization acknowledgment" ("SSYN ACK") **822** to the client **801**. This SSYN ACK **822** will contain the transmit and receive hopblocks that the client **801** will use when communicating with the TARP router **811**. The client **801** will acknowledge the TARP router's **811** response packet **822** by generating an encrypted SSYN ACK ACK packet **823** which will be sent from the client's **801** fixed IP address and to the TARP router's **811** known fixed IP address. The client **801** will simultaneously generate a SSYN ACK ACK packet; this SSYN ACK packet, referred to as the Secure Session Initiation (SSI) packet **824**, will be sent with the first {sender, receiver} IP pair in the client's transmit table **921** (FIG. **9**), as specified in the transmit hopblock provided by the TARP router **811** in the SSYN ACK packet **822**. The TARP router **811** will respond to the SSI packet **824** with an SSI ACK packet **825**, which will be sent with the first {sender, receiver} IP pair in the TARP router's transmit table **923**. Once these packets have been successfully exchanged, the secure communications session is established, and all further secure communications between the client **801** and the TARP router **811** will be conducted via this secure session, as long as synchronization is maintained. If synchronization is lost, then the client **801** and TARP router **802** may re-establish the secure session by the procedure outlined in FIG. **8** and described above.

While the secure session is active, both the client **901** and TARP router **911** (FIG. **9**) will maintain their respective transmit tables **921**, **923** and receive tables **922**, **924**, as provided by the TARP router during session synchronization **822**. It is important that the sequence of IP pairs in the client's transmit table **921** be identical to those in the TARP router's receive table **924**; similarly, the sequence of IP pairs in the client's receive table **922** must be identical to those in the router's transmit table **923**. This is required for the session synchronization to be maintained. The client **901** need maintain only one transmit table **921** and one receive table **922** during the course of the secure session. Each

sequential packet sent by the client **901** will employ the next {send, receive} IP address pair in the transmit table, regardless of TCP or UDP session. The TARP router **911** will expect each packet arriving from the client **901** to bear the next IP address pair shown in its receive table.

Since packets can arrive out of order, however, the router **911** can maintain a "look ahead" buffer in its receive table, and will mark previously-received IP pairs as invalid for future packets; any future packet containing an IP pair that is in the look-ahead buffer but is marked as previously received will be discarded. Communications from the TARP router **911** to the client **901** are maintained in an identical manner; in particular, the router **911** will select the next IP address pair from its transmit table **923** when constructing a packet to send to the client **901**, and the client **901** will maintain a look-ahead buffer of expected IP pairs on packets that it is receiving. Each TARP router will maintain separate pairs of transmit and receive tables for each client that is currently engaged in a secure session with or through that TARP router.

While clients receive their hopblocks from the first server linking them to the Internet, routers exchange hopblocks. When a router establishes a link-based IP-hopping communication regime with another router, each router of the pair exchanges its transmit hopblock. The transmit hopblock of each router becomes the receive hopblock of the other router. The communication between routers is governed as described by the example of a client sending a packet to the first router.

While the above strategy works fine in the IP milieu, many local networks that are connected to the Internet are Ethernet systems. In Ethernet, the IP addresses of the destination devices must be translated into hardware addresses, and vice versa, using known processes ("address resolution protocol," and "reverse address resolution protocol"). However, if the link-based IP-hopping strategy is employed, the correlation process would become explosive and burdensome. An alternative to the link-based IP hopping strategy may be employed within an Ethernet network. The solution is to provide that the node linking the Internet to the Ethernet (call it the border node) use the link-based IP-hopping communication regime to communicate with nodes outside the Ethernet LAN. Within the Ethernet LAN, each TARP node would have a single IP address which would be addressed in the conventional way. Instead of comparing the {sender, receiver} IP address pairs to authenticate a packet, the intra-LAN TARP node would use one of the IP header extension fields to do so. Thus, the border node uses an algorithm shared by the intra-LAN TARP node to generate a symbol that is stored in the free field in the IP header, and the intra-LAN TARP node generates a range of symbols based on its prediction of the next expected packet to be received from that particular source IP address. The packet is rejected if it does not fall into the set of predicted symbols (for example, numerical values) or is accepted if it does. Communications from the intra-LAN TARP node to the border node are accomplished in the same manner, though the algorithm will necessarily be different for security reasons. Thus, each of the communicating nodes will generate transmit and receive tables in a similar manner to that of FIG. **9**; the intra-LAN TARP nodes transmit table will be identical to the border node's receive table, and the intra-LAN TARP node's receive table will be identical to the border node's transmit table.

The algorithm used for IP address-hopping can be any desired algorithm. For example, the algorithm can be a given pseudo-random number generator that generates numbers of the range covering the allowed IP addresses with a given seed. Alternatively, the session participants can assume a certain type of algorithm and specify simply a parameter for applying the algorithm. For example the assumed algorithm could be a particular pseudo-random number generator and the session participants could simply exchange seed values.

Note that there is no permanent physical distinction between the originating and destination terminal nodes. Either device at either end point can initiate a synchronization of the pair. Note also that the authentication/synchronization-request (and acknowledgment) and hopblock-exchange may all be served by a single message so that separate message exchanges may not be required.

As another extension to the stated architecture, multiple physical paths can be used by a client, in order to provide link redundancy and further thwart attempts at denial of service and traffic monitoring. As shown in FIG. **10**, for example, client **1001** can establish three simultaneous sessions with each of three TARP routers provided by different ISPs **1011**, **1012**, **1013**. As an example, the client **1001** can use three different telephone lines **1021**, **1022**, **1023** to connect to the ISPs, or two telephone lines and a cable modem, etc. In this scheme, transmitted packets will be sent in a random fashion among the different physical paths. This architecture provides a high degree of communications redundancy, with improved immunity from denial-of-service attacks and traffic monitoring.

2. FURTHER EXTENSIONS

The following describes various extensions to the techniques, systems, and methods described above. As described above, the security of communications occurring between computers in a computer network (such as the Internet, an Ethernet, or others) can be enhanced by using seemingly random source and destination Internet Protocol (IP) addresses for data packets transmitted over the network. This feature prevents eavesdroppers from determining which computers in the network are communicating with each other while permitting the two communicating computers to easily recognize whether a given received data packet is legitimate or not. In one embodiment of the above-described systems, an IP header extension field is used to authenticate incoming packets on an Ethernet.

Various extensions to the previously described techniques described herein include: (1) use of hopped hardware or "MAC" addresses in broadcast type network; (2) a self-synchronization technique that permits a computer to automatically regain synchronization with a sender; (3) synchronization algorithms that allow transmitting and receiving computers to quickly re-establish synchronization in the event of lost packets or other events; and (4) a fast-packet rejection mechanism for rejecting invalid packets. Any or all of these extensions can be combined with the features described above in any of various ways.

A. Hardware Address Hopping

Internet protocol-based communications techniques on a LAN—or across any dedicated physical medium—typically embed the IP packets within lower-level packets, often referred to as "frames." As shown in FIG. **11**, for example, a first Ethernet frame **1150** comprises a frame header **1101** and two embedded IP packets IP**1** and IP**2**, while a second Ethernet frame **1160** comprises a different frame header **1104** and a single IP packet IP**3**. Each frame header generally includes a source hardware address **1101**A and a destination hardware address **1101**B; other well-known fields in frame headers are omitted from FIG. **11** for clarity. Two

hardware nodes communicating over a physical communication channel insert appropriate source and destination hardware addresses to indicate which nodes on the channel or network should receive the frame.

It may be possible for a nefarious listener to acquire information about the contents of a frame and/or its communicants by examining frames on a local network rather than (or in addition to) the IP packets themselves. This is especially true in broadcast media, such as Ethernet, where it is necessary to insert into the frame header the hardware address of the machine that generated the frame and the hardware address of the machine to which frame is being sent. All nodes on the network can potentially "see" all packets transmitted across the network. This can be a problem for secure communications, especially in cases where the communicants do not want for any third party to be able to identify who is engaging in the information exchange. One way to address this problem is to push the address-hopping scheme down to the hardware layer. In accordance with various embodiments of the invention, hardware addresses are "hopped" in a manner similar to that used to change IP addresses, such that a listener cannot determine which hardware node generated a particular message nor which node is the intended recipient.

FIG. **12**A shows a system in which Media Access Control ("MAC") hardware addresses are "hopped" in order to increase security over a network such as an Ethernet. While the description refers to the exemplary case of an Ethernet environment, the inventive principles are equally applicable to other types of communications media. In the Ethernet case, the MAC address of the sender and receiver are inserted into the Ethernet frame and can be observed by anyone on the LAN who is within the broadcast range for that frame. For secure communications, it becomes desirable to generate frames with MAC addresses that are not attributable to any specific sender or receiver.

As shown in FIG. **12**A, two computer nodes **1201** and **1202** communicate over a communication channel such as an Ethernet. Each node executes one or more application programs **1203** and **1218** that communicate by transmitting packets through communication software **1204** and **1217**, respectively. Examples of application programs include video conferencing, e-mail, word processing programs, telephony, and the like. Communication software **1204** and **1217** can comprise, for example, an OSI layered architecture or "stack" that standardizes various services provided at different levels of functionality.

The lowest levels of communication software **1204** and **1217** communicate with hardware components **1206** and **1214** respectively, each of which can include one or more registers **1207** and **1215** that allow the hardware to be reconfigured or controlled in accordance with various communication protocols. The hardware components (an Ethernet network interface card, for example) communicate with each other over the communication medium. Each hardware component is typically pre-assigned a fixed hardware address or MAC number that identifies the hardware component to other nodes on the network. One or more interface drivers control the operation of each card and can, for example, be configured to accept or reject packets from certain hardware addresses. As will be described in more detail below, various embodiments of the inventive principles provide for "hopping" different addresses using one or more algorithms and one or more moving windows that track a range of valid addresses to validate received packets. Packets transmitted according to one or more of the inventive principles will be generally referred to as "secure" packets or "secure communications" to differentiate them from ordinary data packets that are transmitted in the clear using ordinary, machine-correlated addresses.

One straightforward method of generating non-attributable MAC addresses is an extension of the IP hopping scheme. In this scenario, two machines on the same LAN that desire to communicate in a secure fashion exchange random-number generators and seeds, and create sequences of quasi-random MAC addresses for synchronized hopping. The implementation and synchronization issues are then similar to that of IP hopping.

This approach, however, runs the risk of using MAC addresses that are currently active on the LAN—which, in turn, could interrupt communications for those machines. Since an Ethernet MAC address is at present 48 bits in length, the chance of randomly misusing an active MAC address is actually quite small. However, if that figure is multiplied by a large number of nodes (as would be found on an extensive LAN), by a large number of frames (as might be the case with packet voice or streaming video), and by a large number of concurrent Virtual Private Networks (VPNs), then the chance that a non-secure machine's MAC address could be used in an address-hopped frame can become non-trivial. In short, any scheme that runs even a small risk of interrupting communications for other machines on the LAN is bound to receive resistance from prospective system administrators. Nevertheless, it is technically feasible, and can be implemented without risk on a LAN on which there is a small number of machines, or if all of the machines on the LAN are engaging in MAC-hopped communications.

Synchronized MAC address hopping may incur some overhead in the course of session establishment, especially if there are multiple sessions or multiple nodes involved in the communications. A simpler method of randomizing MAC addresses is to allow each node to receive and process every incident frame on the network. Typically, each network interface driver will check the destination MAC address in the header of every incident frame to see if it matches that machine's MAC address; if there is no match, then the frame is discarded. In one embodiment, however, these checks can be disabled, and every incident packet is passed to the TARP stack for processing. This will be referred to as "promiscuous" mode, since every incident frame is processed. Promiscuous mode allows the sender to use completely random, unsynchronized MAC addresses, since the destination machine is guaranteed to process the frame. The decision as to whether the packet was truly intended for that machine is handled by the TARP stack, which checks the source and destination IP addresses for a match in its IP synchronization tables. If no match is found, the packet is discarded; if there is a match, the packet is unwrapped, the inner header is evaluated, and if the inner header indicates that the packet is destined for that machine then the packet is forwarded to the IP stack—otherwise it is discarded.

One disadvantage of purely-random MAC address hopping is its impact on processing overhead; that is, since every incident frame must be processed, the machine's CPU is engaged considerably more often than if the network interface driver is discriminating and rejecting packets unilaterally. A compromise approach is to select either a single fixed MAC address or a small number of MAC addresses (e.g., one for each virtual private network on an Ethernet) to use for MAC-hopped communications, regardless of the actual recipient for which the message is intended. In this mode, the network interface driver can check each incident

frame against one (or a few) pre-established MAC addresses, thereby freeing the CPU from the task of physical-layer packet discrimination. This scheme does not betray any useful information to an interloper on the LAN; in particular, every secure packet can already be identified by a unique packet type in the outer header. However, since all machines engaged in secure communications would either be using the same MAC address, or be selecting from a small pool of predetermined MAC addresses, the association between a specific machine and a specific MAC address is effectively broken.

In this scheme, the CPU will be engaged more often than it would be in non-secure communications (or in synchronized MAC address hopping), since the network interface driver cannot always unilaterally discriminate between secure packets that are destined for that machine, and secure packets from other VPNs. However, the non-secure traffic is easily eliminated at the network interface, thereby reducing the amount of processing required of the CPU. There are boundary conditions where these statements would not hold, of course-e.g., if all of the traffic on the LAN is secure traffic, then the CPU would be engaged to the same degree as it is in the purely-random address hopping case; alternatively, if each VPN on the LAN uses a different MAC address, then the network interface can perfectly discriminate secure frames destined for the local machine from those constituting other VPNs. These are engineering tradeoffs that might be best handled by providing administrative options for the users when installing the software and/or establishing VPNs.

Even in this scenario, however, there still remains a slight risk of selecting MAC addresses that are being used by one or more nodes on the LAN. One solution to this problem is to formally assign one address or a range of addresses for use in MAC-hopped communications. This is typically done via an assigned numbers registration authority; e.g., in the case of Ethernet, MAC address ranges are assigned to vendors by the Institute of Electrical and Electronics Engineers (IEEE). A formally-assigned range of addresses would ensure that secure frames do not conflict with any properly-configured and properly-functioning machines on the LAN.

Reference will now be made to FIGS. **12**A and **12**B in order to describe the many combinations and features that follow the inventive principles. As explained above, two computer nodes **1201** and **1202** are assumed to be communicating over a network or communication medium such as an Ethernet. A communication protocol in each node (**1204** and **1217**, respectively) contains a modified element **1205** and **1216** that performs certain functions that deviate from the standard communication protocols. In particular, computer node **1201** implements a first "hop" algorithm **1208**X that selects seemingly random source and destination IP addresses (and, in one embodiment, seemingly random IP header discriminator fields) in order to transmit each packet to the other computer node. For example, node **1201** maintains a transmit table **1208** containing triplets of source (S), destination (D), and discriminator fields (DS) that are inserted into outgoing IP packet headers. The table is generated through the use of an appropriate algorithm (e.g., a random number generator that is seeded with an appropriate seed) that is known to the recipient node **1202**. As each new IP packet is formed, the next sequential entry out of the sender's transmit table **1208** is used to populate the IP source, IP destination, and IP header extension field (e.g., discriminator field). It will be appreciated that the transmit table need not be created in advance but could instead be created on-the-fly by executing the algorithm when each packet is formed.

At the receiving node **1202**, the same IP hop algorithm **1222**X is maintained and used to generate a receive table **1222** that lists valid triplets of source IP address, destination IP address, and discriminator field. This is shown by virtue of the first five entries of transmit table **1208** matching the second five entries of receive table **1222**. (The tables may be slightly offset at any particular time due to lost packets, misordered packets, or transmission delays). Additionally, node **1202** maintains a receive window **W3** that represents a list of valid IP source, IP destination, and discriminator fields that will be accepted when received as part of an incoming IP packet. As packets are received, window **W3** slides down the list of valid entries, such that the possible valid entries change over time. Two packets that arrive out of order but are nevertheless matched to entries within window **W3** will be accepted; those falling outside of window **W3** will be rejected as invalid. The length of window **W3** can be adjusted as necessary to reflect network delays or other factors.

Node **1202** maintains a similar transmit table **1221** for creating IP packets and frames destined for node **1201** using a potentially different hopping algorithm **1221**X, and node **1201** maintains a matching receive table **1209** using the same algorithm **1209**X. As node **1202** transmits packets to node **1201** using seemingly random IP source, IP destination, and/or discriminator fields, node **1201** matches the incoming packet values to those falling within window WI maintained in its receive table. In effect, transmit table **1208** of node **1201** is synchronized (i.e., entries are selected in the same order) to receive table **1222** of receiving node **1202**. Similarly, transmit table **1221** of node **1202** is synchronized to receive table **1209** of node **1201**. It will be appreciated that although a common algorithm is shown for the source, destination and discriminator fields in FIG. **12**A (using, e.g., a different seed for each of the three fields), an entirely different algorithm could in fact be used to establish values for each of these fields. It will also be appreciated that one or two of the fields can be "hopped" rather than all three as illustrated.

In accordance with another aspect of the invention, hardware or "MAC" addresses are hopped instead of or in addition to IP addresses and/or the discriminator field in order to improve security in a local area or broadcast-type network. To that end, node **1201** further maintains a transmit table **1210** using a transmit algorithm **1210**X to generate source and destination hardware addresses that are inserted into frame headers (e.g., fields **1101**A and **1101**B in FIG. **11**) that are synchronized to a corresponding receive table **1224** at node **1202**. Similarly, node **1202** maintains a different transmit table **1223** containing source and destination hardware addresses that is synchronized with a corresponding receive table **1211** at node **1201**. In this manner, outgoing hardware frames appear to be originating from and going to completely random nodes on the network, even though each recipient can determine whether a given packet is intended for it or not. It will be appreciated that the hardware hopping feature can be implemented at a different level in the communications protocol than the IP hopping feature (e.g., in a card driver or in a hardware card itself to improve performance).

FIG. **12**B shows three different embodiments or modes that can be employed using the aforementioned principles. In a first mode referred to as "promiscuous" mode, a common hardware address (e.g., a fixed address for source and another for destination) or else a completely random hardware address is used by all nodes on the network, such that a particular packet cannot be attributed to any one node.

Each node must initially accept all packets containing the common (or random) hardware address and inspect the IP addresses or discriminator field to determine whether the packet is intended for that node. In this regard, either the IP addresses or the discriminator field or both can be varied in accordance with an algorithm as described above. As explained previously, this may increase each node's overhead since additional processing is involved to determine whether a given packet has valid source and destination hardware addresses.

In a second mode referred to as "promiscuous per VPN" mode, a small set of fixed hardware addresses are used, with a fixed source/destination hardware address used for all nodes communicating over a virtual private network. For example, if there are six nodes on an Ethernet, and the network is to be split up into two private virtual networks such that nodes on one VPN can communicate with only the other two nodes on its own VPN, then two sets of hardware addresses could be used: one set for the first VPN and a second set for the second VPN. This would reduce the amount of overhead involved in checking for valid frames since only packets arriving from the designated VPN would need to be checked. IP addresses and one or more discriminator fields could still be hopped as before for secure communication within the VPN. Of course, this solution compromises the anonymity of the VPNs (i.e., an outsider can easily tell what traffic belongs in which VPN, though he cannot correlate it to a specific machine/person). It also requires the use of a discriminator field to mitigate the vulnerability to certain types of DoS attacks. (For example, without the discriminator field, an attacker on the LAN could stream frames containing the MAC addresses being used by the VPN; rejecting those frames could lead to excessive processing overhead. The discriminator field would provide a low-overhead means of rejecting the false packets.)

In a third mode referred to as "hardware hopping" mode, hardware addresses are varied as illustrated in FIG. **12**A, such that hardware source and destination addresses are changed constantly in order to provide non-attributable addressing. Variations on these embodiments are of course possible, and the invention is not intended to be limited in any respect by these illustrative examples.

B. Extending the Address Space

Address hopping provides security and privacy. However, the level of protection is limited by the number of addresses in the blocks being hopped. A hopblock denotes a field or fields modulated on a packet-wise basis for the purpose of providing a VPN. For instance, if two nodes communicate with IP address hopping using hopblocks of 4 addresses (2 bits) each, there would be 16 possible address-pair combinations. A window of size 16 would result in most address pairs being accepted as valid most of the time. This limitation can be overcome by using a discriminator field in addition to or instead of the hopped address fields. The discriminator field would be hopped in exactly the same fashion as the address fields and it would be used to determine whether a packet should be processed by a receiver.

Suppose that two clients, each using four-bit hopblocks, would like the same level of protection afforded to clients communicating via IP hopping between two A blocks (24 address bits eligible for hopping). A discriminator field of 20 bits, used in conjunction with the 4 address bits eligible for hopping in the IP address field, provides this level of protection. A 24-bit discriminator field would provide a similar level of protection if the address fields were not hopped or ignored. Using a discriminator field offers the following advantages: (1) an arbitrarily high level of protection can be provided, and (2) address hopping is unnecessary to provide protection. This may be important in environments where address hopping would cause routing problems.

C. Synchronization Techniques

It is generally assumed that once a sending node and receiving node have exchanged algorithms and seeds (or similar information sufficient to generate quasi-random source and destination tables), subsequent communication between the two nodes will proceed smoothly. Realistically, however, two nodes may lose synchronization due to network delays or outages, or other problems. Consequently, it is desirable to provide means for re-establishing synchronization between nodes in a network that have lost synchronization.

One possible technique is to require that each node provide an acknowledgment upon successful receipt of each packet and, if no acknowledgment is received within a certain period of time, to re-send the unacknowledged packet. This approach, however, drives up overhead costs and may be prohibitive in high-throughput environments such as streaming video or audio, for example.

A different approach is to employ an automatic synchronizing technique that will be referred to herein as "self-synchronization." In this approach, synchronization information is embedded into each packet, thereby enabling the receiver to re-synchronize itself upon receipt of a single packet if it determines that is has lost synchronization with the sender. (If communications are already in progress, and the receiver determines that it is still in sync with the sender, then there is no need to re-synchronize.) A receiver could detect that it was out of synchronization by, for example, employing a "dead-man" timer that expires after a certain period of time, wherein the timer is reset with each valid packet. A time stamp could be hashed into the public sync field (see below) to preclude packet-retry attacks.

In one embodiment, a "sync field" is added to the header of each packet sent out by the sender. This sync field could appear in the clear or as part of an encrypted portion of the packet. Assuming that a sender and receiver have selected a random-number generator (RNG) and seed value, this combination of RNG and seed can be used to generate a random-number sequence (RNS). The RNS is then used to generate a sequence of source/destination IP pairs (and, if desired, discriminator fields and hardware source and destination addresses), as described above. It is not necessary, however, to generate the entire sequence (or the first N−1 values) in order to generate the Nth random number in the sequence; if the sequence index N is known, the random value corresponding to that index can be directly generated (see below). Different RNGs (and seeds) with different fundamental periods could be used to generate the source and destination IP sequences, but the basic concepts would still apply. For the sake of simplicity, the following discussion will assume that IP source and destination address pairs (only) are hopped using a single RNG sequencing mechanism.

In accordance with a "self-synchronization" feature, a sync field in each packet header provides an index (i.e., a sequence number) into the RNS that is being used to generate IP pairs. Plugging this index into the RNG that is being used to generate the RNS yields a specific random number value, which in turn yields a specific IP pair. That is, an IP pair can be generated directly from knowledge of the RNG, seed, and index number; it is not necessary, in this

scheme, to generate the entire sequence of random numbers that precede the sequence value associated with the index number provided.

Since the communicants have presumably previously exchanged RNGs and seeds, the only new information that must be provided in order to generate an IP pair is the sequence number. If this number is provided by the sender in the packet header, then the receiver need only plug this number into the RNG in order to generate an IP pair—and thus verify that the IP pair appearing in the header of the packet is valid. In this scheme, if the sender and receiver lose synchronization, the receiver can immediately re-synchronize upon receipt of a single packet by simply comparing the IP pair in the packet header to the IP pair generated from the index number. Thus, synchronized communications can be resumed upon receipt of a single packet, making this scheme ideal for multicast communications. Taken to the extreme, it could obviate the need for synchronization tables entirely; that is, the sender and receiver could simply rely on the index number in the sync field to validate the IP pair on each packet, and thereby eliminate the tables entirely.

The aforementioned scheme may have some inherent security issues associated with it—namely, the placement of the sync field. If the field is placed in the outer header, then an interloper could observe the values of the field and their relationship to the IP stream. This could potentially compromise the algorithm that is being used to generate the IP-address sequence, which would compromise the security of the communications. If, however, the value is placed in the inner header, then the sender must decrypt the inner header before it can extract the sync value and validate the IP pair; this opens up the receiver to certain types of denial-of-service (DoS) attacks, such as packet replay. That is, if the receiver must decrypt a packet before it can validate the IP pair, then it could potentially be forced to expend a significant amount of processing on decryption if an attacker simply retransmits previously valid packets. Other attack methodologies are possible in this scenario.

A possible compromise between algorithm security and processing speed is to split up the sync value between an inner (encrypted) and outer (unencrypted) header. That is, if the sync value is sufficiently long, it could potentially be split into a rapidly-changing part that can be viewed in the clear, and a fixed (or very slowly changing) part that must be protected. The part that can be viewed in the clear will be called the "public sync" portion and the part that must be protected will be called the "private sync" portion.

Both the public sync and private sync portions are needed to generate the complete sync value. The private portion, however, can be selected such that it is fixed or will change only occasionally. Thus, the private sync value can be stored by the recipient, thereby obviating the need to decrypt the header in order to retrieve it. If the sender and receiver have previously agreed upon the frequency with which the private part of the sync will change, then the receiver can selectively decrypt a single header in order to extract the new private sync if the communications gap that has led to lost synchronization has exceeded the lifetime of the previous private sync. This should not represent a burdensome amount of decryption, and thus should not open up the receiver to denial-of-service attack simply based on the need to occasionally decrypt a single header.

One implementation of this is to use a hashing function with a one-to-one mapping to generate the private and public sync portions from the sync value. This implementation is shown in FIG. **13**, where (for example) a first ISP **1302** is the sender and a second ISP **1303** is the receiver. (Other alternatives are possible from FIG. **13**.) A transmitted packet comprises a public or "outer" header **1305** that is not encrypted, and a private or "inner" header **1306** that is encrypted using for example a link key. Outer header **1305** includes a public sync portion while inner header **1306** contains the private sync portion. A receiving node decrypts the inner header using a decryption function **1307** in order to extract the private sync portion. This step is necessary only if the lifetime of the currently buffered private sync has expired. (If the currently-buffered private sync is still valid, then it is simply extracted from memory and "added" (which could be an inverse hash) to the public sync, as shown in step **1308**.) The public and decrypted private sync portions are combined in function **1308** in order to generate the combined sync **1309**. The combined sync (**1309**) is then fed into the RNG (**1310**) and compared to the IP address pair (**1311**) to validate or reject the packet.

An important consideration in this architecture is the concept of "future" and "past" where the public sync values are concerned. Though the sync values, themselves, should be random to prevent spoofing attacks, it may be important that the receiver be able to quickly identify a sync value that has already been sent—even if the packet containing that sync value was never actually received by the receiver. One solution is to hash a time stamp or sequence number into the public sync portion, which could be quickly extracted, checked, and discarded, thereby validating the public sync portion itself.

In one embodiment, packets can be checked by comparing the source/destination IP pair generated by the sync field with the pair appearing in the packet header. If (1) they match, (2) the time stamp is valid, and (3) the dead-man timer has expired, then re-synchronization occurs; otherwise, the packet is rejected. If enough processing power is available, the dead-man timer and synchronization tables can be avoided altogether, and the receiver would simply resynchronize (e.g., validate) on every packet.

The foregoing scheme may require large-integer (e.g., 160-bit) math, which may affect its implementation. Without such large-integer registers, processing throughput would be affected, thus potentially affecting security from a denial-of-service standpoint. Nevertheless, as large-integer math processing features become more prevalent, the costs of implementing such a feature will be reduced.

D. Other Synchronization Schemes

As explained above, if W or more consecutive packets are lost between a transmitter and receiver in a VPN (where W is the window size), the receiver's window will not have been updated and the transmitter will be transmitting packets not in the receiver's window. The sender and receiver will not recover synchronization until perhaps the random pairs in the window are repeated by chance. Therefore, there is a need to keep a transmitter and receiver in synchronization whenever possible and to re-establish synchronization whenever it is lost.

A "checkpoint" scheme can be used to regain synchronization between a sender and a receiver that have fallen out of synchronization. In this scheme, a checkpoint message comprising a random IP address pair is used for communicating synchronization information. In one embodiment, two messages are used to communicate synchronization information between a sender and a recipient:

1. SYNC_REQ is a message used by the sender to indicate that it wants to synchronize; and
2. SYNC_ACK is a message used by the receiver to inform the transmitter that it has been synchronized.

According to one variation of this approach, both the transmitter and receiver maintain three checkpoints (see FIG. **14**):

1. In the transmitter, ckpt_o ("checkpoint old") is the IP pair that was used to re-send the last SYNC_REQ packet to the receiver. In the receiver, ckpt_o ("checkpoint old") is the IP pair that receives repeated SYNC_REQ packets from the transmitter.
2. In the transmitter, ckpt_n ("checkpoint new") is the IP pair that will be used to send the next SYNC_REQ packet to the receiver. In the receiver, ckpt_n ("checkpoint new") is the IP pair that receives a new SYNC_REQ packet from the transmitter and which causes the receiver's window to be re-aligned, ckpt_o set to ckpt_n, a new ckpt_n to be generated and a new ckpt_r to be generated.
3. In the transmitter, ckpt_r is the IP pair that will be used to send the next SYNC_ACK packet to the receiver. In the receiver, ckpt_r is the IP pair that receives a new SYNC_ACK packet from the transmitter and which causes a new ckpt_n to be generated. Since SYNC_ACK is transmitted from the receiver ISP to the sender ISP, the transmitter ckpt_r refers to the ckpt_r of the receiver and the receiver ckpt_r refers to the ckpt_r of the transmitter (see FIG. **14**).

When a transmitter initiates synchronization, the IP pair it will use to transmit the next data packet is set to a predetermined value and when a receiver first receives a SYNC_REQ, the receiver window is updated to be centered on the transmitter's next IP pair. This is the primary mechanism for checkpoint synchronization.

Synchronization can be initiated by a packet counter (e.g., after every N packets transmitted, initiate a synchronization) or by a timer (every S seconds, initiate a synchronization) or a combination of both. See FIG. **15**. From the transmitter's perspective, this technique operates as follows: (1) Each transmitter periodically transmits a "sync request" message to the receiver to make sure that it is in sync. (2) If the receiver is still in sync, it sends back a "sync ack" message. (If this works, no further action is necessary). (3) If no "sync ack" has been received within a period of time, the transmitter retransmits the sync request again. If the transmitter reaches the next checkpoint without receiving a "sync ack" response, then synchronization is broken, and the transmitter should stop transmitting. The transmitter will continue to send sync_reqs until it receives a sync_ack, at which point transmission is reestablished.

From the receiver's perspective, the scheme operates as follows: (1) when it receives a "sync request" request from the transmitter, it advances its window to the next checkpoint position (even skipping pairs if necessary), and sends a "sync ack" message to the transmitter. If sync was never lost, then the "jump ahead" really just advances to the next available pair of addresses in the table (i.e., normal advancement).

If an interloper intercepts the "sync request" messages and tries to interfere with communication by sending new ones, it will be ignored if the synchronization has been established or it it will actually help to re-establish synchronization.

A window is realigned whenever a re-synchronization occurs. This realignment entails updating the receiver's window to straddle the address pairs used by the packet transmitted immediately after the transmission of the SYNC_REQ packet. Normally, the transmitter and receiver are in synchronization with one another. However, when network events occur, the receiver's window may have to be advanced by many steps during resynchronization. In this case, it is desirable to move the window ahead without having to step through the intervening random numbers sequentially. (This feature is also desirable for the auto-sync approach discussed above).

E. Random Number Generator with a Jump-Ahead Capability

An attractive method for generating randomly hopped addresses is to use identical random number generators in the transmitter and receiver and advance them as packets are transmitted and received. There are many random number generation algorithms that could be used. Each one has strengths and weaknesses for address hopping applications.

Linear congruential random number generators (LCRs) are fast, simple and well characterized random number generators that can be made to jump ahead n steps efficiently. An LCR generates random numbers X_1, X_2, X_3 . . . X_k starting with seed X_0 using a recurrence

$$X_i=(aX_{i-1}+b)\bmod c, \tag{1}$$

where a, b and c define a particular LCR. Another expression for X_i,

$$X_i=((a^i(X_0+b)-b)/(a-1))\bmod c \tag{2}$$

enables the jump-ahead capability. The factor a^i can grow very large even for modest i if left unfettered. Therefore some special properties of the modulo operation can be used to control the size and processing time required to compute (2). (2) can be rewritten as:

$$X_i=(a^i(X_0(a-1)+b)-b)/(a-1)\bmod c. \tag{3}$$

It can be shown that:

$$(a^i(X_0(a-1)+b)-b)/(a-1)\bmod c=((a^i \bmod((a-1)c)(X_0(a-1)+b)-b)/(a-1))\bmod c \tag{4}.$$

$(X_0(a-1)+b)$ can be stored as $(X_0(a-1)+b) \bmod c$, b as b mod c and compute $a^i \bmod((a-1)c)$ (this requires O(log(i)) steps).

A practical implementation of this algorithm would jump a fixed distance, n, between synchronizations; this is tantamount to synchronizing every n packets. The window would commence n IP pairs from the start of the previous window. Using X_j^w, the random number at the j^{th} checkpoint, as X_0 and n as i, a node can store $a^n \bmod((a-1)c)$ once per LCR and set

$$X_{j+1}^w=X_{n(j+1)}=((a^n \bmod((a-1)c)(X_j^w(a-1)+b)-b)/(a-1))\bmod c, \tag{5}$$

to generate the random number for the $j+1^{th}$ synchronization. Using this construction, a node could jump ahead an arbitrary (but fixed) distance between synchronizations in a constant amount of time (independent of n).

Pseudo-random number generators, in general, and LCRs, in particular, will eventually repeat their cycles. This repetition may present vulnerability in the IP hopping scheme. An adversary would simply have to wait for a repeat to predict future sequences. One way of coping with this vulnerability is to create a random number generator with a known long cycle. A random sequence can be replaced by a new random number generator before it repeats. LCRs can be constructed with known long cycles. This is not currently true of many random number generators.

Random number generators can be cryptographically insecure. An adversary can derive the RNG parameters by examining the output or part of the output. This is true of LCGs. This vulnerability can be mitigated by incorporating an encryptor, designed to scramble the output as part of the random number generator. The random number generator

prevents an adversary from mounting an attack—e.g., a known plaintext attack—against the encryptor.

F. Random Number Generator Example

Consider a RNG where a=31, b=4 and c=15. For this case equation (1) becomes:

$$X_i=(31X_{i-1}+4)\bmod 15. \quad (6)$$

If one sets X_0=1, equation (6) will produce the sequence 1, 5, 9, 13, 2, 6, 10, 14, 3, 7, 11, 0, 4, 8, 12. This sequence will repeat indefinitely. For a jump ahead of 3 numbers in this sequence $a^n=31^3=29791$, $c^*(a-1)=15^*30=450$ and $a^n \bmod((a-1)c)=31^3 \bmod(15^*30)=29791 \bmod(450)=91$. Equation (5) becomes:

$$((91(X_i30+4)-4)/30)\bmod 15 \quad (7).$$

Table 1 shows the jump ahead calculations from (7) . The calculations start at 5 and jump ahead 3.

TABLE 1

I	X_i	$(X_i30 + 4)$	91 $(X_i30 + 4) - 4$	$((91$ $(X_i30 + 4) - 4)/30$	X_{i+3}
1	5	154	14010	467	2
4	2	64	5820	194	14
7	14	424	38580	1286	11
10	11	334	30390	1013	8
13	8	244	22200	740	5

G. Fast Packet Filter

Address hopping VPNs must rapidly determine whether a packet has a valid header and thus requires further processing, or has an invalid header (a hostile packet) and should be immediately rejected. Such rapid determinations will be referred to as "fast packet filtering." This capability protects the VPN from attacks by an adversary who streams hostile packets at the receiver at a high rate of speed in the hope of saturating the receiver's processor (a so-called "denial of service" attack). Fast packet filtering is an important feature for implementing VPNs on shared media such as Ethernet.

Assuming that all participants in a VPN share an unassigned "A" block of addresses, one possibility is to use an experimental "A" block that will never be assigned to any machine that is not address hopping on the shared medium. "A" blocks have a 24 bits of address that can be hopped as opposed to the 8 bits in "C" blocks. In this case a hopblock will be the "A" block. The use of the experimental "A" block is a likely option on an Ethernet because:

1. The addresses have no validity outside of the Ethernet and will not be routed out to a valid outside destination by a gateway.
2. There are 2^{24} (~16 million) addresses that can be hopped within each "A" block. This yields >280 trillion possible address pairs making it very unlikely that an adversary would guess a valid address. It also provides acceptably low probability of collision between separate VPNs (all VPNs on a shared medium independently generate random address pairs from the same "A" block).
3. The packets will not be received by someone on the Ethernet who is not on a VPN (unless the machine is in promiscuous mode) minimizing impact on non-VPN computers.

The Ethernet example will be used to describe one implementation of fast packet filtering. The ideal algorithm would quickly examine a packet header, determine whether the packet is hostile, and reject any hostile packets or determine which active IP pair the packet header matches. The problem is a classical associative memory problem. A variety of techniques have been developed to solve this problem (hashing, B-trees etc). Each of these approaches has its strengths and weaknesses. For instance, hash tables can be made to operate quite fast in a statistical sense, but can occasionally degenerate into a much slower algorithm. This slowness can persist for a period of time. Since there is a need to discard hostile packets quickly at all times, hashing would be unacceptable.

H. Presence Vector Algorithm

A presence vector is a bit vector of length 2^n that can be indexed by n-bit numbers (each ranging from 0 to 2^n-1). One can indicate the presence of k n-bit numbers (not necessarily unique), by setting the bits in the presence vector indexed by each number to 1. Otherwise, the bits in the presence vector are 0. An n-bit number, x, is one of the k numbers if and only if the x^{th} bit of the presence vector is 1. A fast packet filter can be implemented by indexing the presence vector and looking for a 1, which will be referred to as the "test."

For example, suppose one wanted to represent the number **135** using a presence vector. The 135^{th} bit of the vector would be set. Consequently, one could very quickly determine whether an address of 135 was valid by checking only one bit: the 135^{th} bit. The presence vectors could be created in advance corresponding to the table entries for the IP addresses. In effect, the incoming addresses can be used as indices into a long vector, making comparisons very fast. As each RNG generates a new address, the presence vector is updated to reflect the information. As the window moves, the presence vector is updated to zero out addresses that are no longer valid.

There is a trade-off between efficiency of the test and the amount of memory required for storing the presence vector (s). For instance, if one were to use the 48 bits of hopping addresses as an index, the presence vector would have to be 35 terabytes. Clearly, this is too large for practical purposes. Instead, the 48 bits can be divided into several smaller fields. For instance, one could subdivide the 48 bits into four 12-bit fields (see FIG. **16**). This reduces the storage requirement to 2048 bytes at the expense of occasionally having to process a hostile packet. In effect, instead of one long presence vector, the decomposed address portions must match all four shorter presence vectors before further processing is allowed. (If the first part of the address portion doesn't match the first presence vector, there is no need to check the remaining three presence vectors).

A presence vector will have a 1 in the y^{th} bit if and only if one or more addresses with a corresponding field of y are active. An address is active only if each presence vector indexed by the appropriate sub-field of the address is 1.

Consider a window of 32 active addresses and 3 checkpoints. A hostile packet will be rejected by the indexing of one presence vector more than 99% of the time. A hostile packet will be rejected by the indexing of all 4 presence vectors more than 99.9999995% of the time. On average, hostile packets will be rejected in less than 1.02 presence vector index operations.

The small percentage of hostile packets that pass the fast packet filter will be rejected when matching pairs are not found in the active window or are active checkpoints. Hostile packets that serendipitously match a header will be rejected when the VPN software attempts to decrypt the header. However, these cases will be extremely rare. There are many other ways this method can be configured to arbitrate the space/speed tradeoffs.

I. Further Synchronization Enhancements

A slightly modified form of the synchronization techniques described above can be employed. The basic principles of the previously described checkpoint synchronization scheme remain unchanged. The actions resulting from the reception of the checkpoints are, however, slightly different. In this variation, the receiver will maintain between OoO ("Out of Order") and 2×WINDOW_SIZE+OoO active addresses ($1 \leqq OoO \leqq WINDOW\_SIZE$ and $WINDOW\_SIZE \geqq 1$). OoO and WINDOW_SIZE are engineerable parameters, where OoO is the minimum number of addresses needed to accommodate lost packets due to events in the network or out of order arrivals and WINDOW_SIZE is the number of packets transmitted before a SYNC_REQ is issued. FIG. **17** depicts a storage array for a receiver's active addresses.

The receiver starts with the first 2×WINDOW_SIZE addresses loaded and active (ready to receive data). As packets are received, the corresponding entries are marked as "used" and are no longer eligible to receive packets. The transmitter maintains a packet counter, initially set to 0, containing the number of data packets transmitted since the last initial transmission of a SYNC_REQ for which SYNC_ACK has been received. When the transmitter packet counter equals WINDOW_SIZE, the transmitter generates a SYNC_REQ and does its initial transmission. When the receiver receives a SYNC_REQ corresponding to its current CKPT_N, it generates the next WINDOW_SIZE addresses and starts loading them in order starting at the first location after the last active address wrapping around to the beginning of the array after the end of the array has been reached. The receiver's array might look like FIG. **18** when a SYNC_REQ has been received. In this case a couple of packets have been either lost or will be received out of order when the SYNC_REQ is received.

FIG. **19** shows the receiver's array after the new addresses have been generated. If the transmitter does not receive a SYNC_ACK, it will re-issue the SYNC_REQ at regular intervals. When the transmitter receives a SYNC_ACK, the packet counter is decremented by WINDOW_SIZE. If the packet counter reaches 2×WINDOW_SIZE—OoO then the transmitter ceases sending data packets until the appropriate SYNC_ACK is finally received. The transmitter then resumes sending data packets. Future behavior is essentially a repetition of this initial cycle. The advantages of this approach are:

1. There is no need for an efficient jump ahead in the random number generator,
2. No packet is ever transmitted that does not have a corresponding entry in the receiver side
3. No timer based re-synchronization is necessary. This is a consequence of 2.
4. The receiver will always have the ability to accept data messages transmitted within OoO messages of the most recently transmitted message.

J. Distributed Transmission Path Variant

Another embodiment incorporating various inventive principles is shown in FIG. **20**. In this embodiment, a message transmission system includes a first computer **2001** in communication with a second computer **2002** through a network **2011** of intermediary computers. In one variant of this embodiment, the network includes two edge routers **2003** and **2004** each of which is linked to a plurality of Internet Service Providers (ISPs) **2005** through **2010**. Each ISP is coupled to a plurality of other ISPs in an arrangement as shown in FIG. **20**, which is a representative configuration only and is not intended to be limiting. Each connection between ISPs is labeled in FIG. **20** to indicate a specific physical transmission path (e.g., AD is a physical path that links ISP A (element **2005**) to ISP D (element **2008**)). Packets arriving at each edge router are selectively transmitted to one of the ISPs to which the router is attached on the basis of a randomly or quasi-randomly selected basis.

As shown in FIG. **21**, computer **2001** or edge router **2003** incorporates a plurality of link transmission tables **2100** that identify, for each potential transmission path through the network, valid sets of IP addresses that can be used to transmit the packet. For example, AD table **2101** contains a plurality of IP source/destination pairs that are randomly or quasi-randomly generated. When a packet is to be transmitted from first computer **2001** to second computer **2002**, one of the link tables is randomly (or quasi-randomly) selected, and the next valid source/destination address pair from that table is used to transmit the packet through the network. If path AD is randomly selected, for example, the next source/destination IP address pair (which is pre-determined to transmit between ISP A (element **2005**) and ISP B (element **2008**)) is used to transmit the packet. If one of the transmission paths becomes degraded or inoperative, that link table can be set to a "down" condition as shown in table **2105**, thus preventing addresses from being selected from that table. Other transmission paths would be unaffected by this broken link.

3. CONTINUATION-IN-PART IMPROVEMENTS

The following describes various improvements and features that can be applied to the embodiments described above. The improvements include: (1) a load balancer that distributes packets across different transmission paths according to transmission path quality; (2) a DNS proxy server that transparently creates a virtual private network in response to a domain name inquiry; (3) a large-to-small link bandwidth management feature that prevents denial-of-service attacks at system chokepoints; (4) a traffic limiter that regulates incoming packets by limiting the rate at which a transmitter can be synchronized with a receiver; and (5) a signaling synchronizer that allows a large number of nodes to communicate with a central node by partitioning the communication function between two separate entities. Each is discussed separately below.

A. Load Balancer

Various embodiments described above include a system in which a transmitting node and a receiving node are coupled through a plurality of transmission paths, and wherein successive packets are distributed quasi-randomly over the plurality of paths. See, for example, FIGS. **20** and **21** and accompanying description. The improvement extends this basic concept to encompass distributing packets across different paths in such a manner that the loads on the paths are generally balanced according to transmission link quality.

In one embodiment, a system includes a transmitting node and a receiving node that are linked via a plurality of transmission paths having potentially varying transmission quality. Successive packets are transmitted over the paths based on a weight value distribution function for each path. The rate that packets will be transmitted over a given path can be different for each path. The relative "health" of each transmission path is monitored in order to identify paths that have become degraded. In one embodiment, the health of each path is monitored in the transmitter by comparing the number of packets transmitted to the number of packet acknowledgements received. Each transmission path may comprise a physically separate path (e.g., via dial-up phone

line, computer network, router, bridge, or the like), or may comprise logically separate paths contained within a broadband communication medium (e.g., separate channels in an FDM, TDM, CDMA, or other type of modulated or unmodulated transmission link).

When the transmission quality of a path falls below a predetermined threshold and there are other paths that can transmit packets, the transmitter changes the weight value used for that path, making it less likely that a given packet will be transmitted over that path. The weight will preferably be set no lower than a minimum value that keeps nominal traffic on the path. The weights of the other available paths are altered to compensate for the change in the affected path. When the quality of a path degrades to where the transmitter is turned off by the synchronization function (i.e., no packets are arriving at the destination), the weight is set to zero. If all transmitters are turned off, no packets are sent.

Conventional TCP/IP protocols include a "throttling" feature that reduces the transmission rate of packets when it is determined that delays or errors are occurring in transmission. In this respect, timers are sometimes used to determine whether packets have been received. These conventional techniques for limiting transmission of packets, however, do not involve multiple transmission paths between two nodes wherein transmission across a particular path relative to the others is changed based on link quality.

According to certain embodiments, in order to damp oscillations that might otherwise occur if weight distributions are changed drastically (e.g., according to a step function), a linear or an exponential decay formula can be applied to gradually decrease the weight value over time that a degrading path will be used. Similarly, if the health of a degraded path improves, the weight value for that path is gradually increased.

Transmission link health can be evaluated by comparing the number of packets that are acknowledged within the transmission window (see embodiments discussed above) to the number of packets transmitted within that window and by the state of the transmitter (i.e., on or off). In other words, rather than accumulating general transmission statistics over time for a path, one specific implementation uses the "windowing" concepts described above to evaluate transmission path health.

The same scheme can be used to shift virtual circuit paths from an "unhealthy" path to a "healthy" one, and to select a path for a new virtual circuit.

FIG. **22**A shows a flowchart for adjusting weight values associated with a plurality of transmission links. It is assumed that software executing in one or more computer nodes executes the steps shown in FIG. **22**A. It is also assumed that the software can be stored on a computer-readable medium such as a magnetic or optical disk for execution by a computer.

Beginning in step **2201**, the transmission quality of a given transmission path is measured. As described above, this measurement can be based on a comparison between the number of packets transmitted over a particular link to the number of packet acknowledgements received over the link (e.g., per unit time, or in absolute terms). Alternatively, the quality can be evaluated by comparing the number of packets that are acknowledged within the transmission window to the number of packets that were transmitted within that window. In yet another variation, the number of missed synchronization messages can be used to indicate link quality. Many other variations are of course possible.

In step **2202**, a check is made to determine whether more than one transmitter (e.g., transmission path) is turned on. If not, the process is terminated and resumes at step **2201**.

In step **2203**, the link quality is compared to a given threshold (e.g., 50%, or any arbitrary number). If the quality falls below the threshold, then in step **2207** a check is made to determine whether the weight is above a minimum level (e.g., 1%). If not, then in step **2209** the weight is set to the minimum level and processing resumes at step **2201**. If the weight is above the minimum level, then in step **2208** the weight is gradually decreased for the path, then in step **2206** the weights for the remaining paths are adjusted accordingly to compensate (e.g., they are increased).

If in step **2203** the quality of the path was greater than or equal to the threshold, then in step **2204** a check is made to determine whether the weight is less than a steady-state value for that path. If so, then in step **2205** the weight is increased toward the steady-state value, and in step **2206** the weights for the remaining paths are adjusted accordingly to compensate (e.g., they are decreased). If in step **2204** the weight is not less than the steady-state value, then processing resumes at step **2201** without adjusting the weights.

The weights can be adjusted incrementally according to various functions, preferably by changing the value gradually. In one embodiment, a linearly decreasing function is used to adjust the weights; according to another embodiment, an exponential decay function is used. Gradually changing the weights helps to damp oscillators that might otherwise occur if the probabilities were abruptly.

Although not explicitly shown in FIG. **22**A the process can be performed only periodically (e.g., according to a time schedule), or it can be continuously run, such as in a background mode of operation. In one embodiment, the combined weights of all potential paths should add up to unity (e.g., when the weighting for one path is decreased, the corresponding weights that the other paths will be selected will increase).

Adjustments to weight values for other paths can be prorated. For example, a decrease of 10% in weight value for one path could result in an evenly distributed increase in the weights for the remaining paths. Alternatively, weightings could be adjusted according to a weighted formula as desired (e.g., favoring healthy paths over less healthy paths). In yet another variation, the difference in weight value can be amortized over the remaining links in a manner that is proportional to their traffic weighting.

FIG. **22**B shows steps that can be executed to shut down transmission links where a transmitter turns off. In step **2210**, a transmitter shut-down event occurs. In step **2211**, a test is made to determine whether at least one transmitter is still turned on. If not, then in step **2215** all packets are dropped until a transmitter turns on. If in step **2211** at least one transmitter is turned on, then in step **2212** the weight for the path is set to zero, and the weights for the remaining paths are adjusted accordingly.

FIG. **23** shows a computer node **2301** employing various principles of the above-described embodiments. It is assumed that two computer nodes of the type shown in FIG. **23** communicate over a plurality of separate physical transmission paths. As shown in FIG. **23**, four transmission paths X**1** through X**4** are defined for communicating between the two nodes. Each node includes a packet transmitter **2302** that operates in accordance with a transmit table **2308** as described above. (The packet transmitter could also operate without using the IP-hopping features described above, but the following description assumes that some form of hopping is employed in conjunction with the path selection mechanism.). The computer node also includes a packet receiver **2303** that operates in accordance with a receive table **2309**, including a moving window W that moves as

valid packets are received. Invalid packets having source and destination addresses that do not fall within window W are rejected.

As each packet is readied for transmission, source and destination IP addresses (or other discriminator values) are selected from transmit table **2308** according to any of the various algorithms described above, and packets containing these source/destination address pairs, which correspond to the node to which the four transmission paths are linked, are generated to a transmission path switch **2307**. Switch **2307**, which can comprise a software function, selects from one of the available transmission paths according to a weight distribution table **2306**. For example, if the weight for path X1 is 0.2, then every fifth packet will be transmitted on path X1. A similar regime holds true for the other paths as shown. Initially, each link's weight value can be set such that it is proportional to its bandwidth, which will be referred to as its "steady-state" value.

Packet receiver **2303** generates an output to a link quality measurement function **2304** that operates as described above to determine the quality of each transmission path. (The input to packet receiver **2303** for receiving incoming packets is omitted for clarity). Link quality measurement function **2304** compares the link quality to a threshold for each transmission link and, if necessary, generates an output to weight adjustment function **2305**. If a weight adjustment is required, then the weights in table **2306** are adjusted accordingly, preferably according to a gradual (e.g., linearly or exponentially declining) function. In one embodiment, the weight values for all available paths are initially set to the same value, and only when paths degrade in quality are the weights changed to reflect differences.

Link quality measurement function **2304** can be made to operate as part of a synchronizer function as described above. That is, if resynchronization occurs and the receiver detects that synchronization has been lost (e.g., resulting in the synchronization window W being advanced out of sequence), that fact can be used to drive link quality measurement function **2304**. According to one embodiment, load balancing is performed using information garnered during the normal synchronization, augmented slightly to communicate link health from the receiver to the transmitter. The receiver maintains a count, MESS_R(W), of the messages received in synchronization window W. When it receives a synchronization request (SYNC_REQ) corresponding to the end of window W, the receiver includes counter MESS_R in the resulting synchronization acknowledgement (SYNC_ACK) sent back to the transmitter. This allows the transmitter to compare messages sent to messages received in order to asses the health of the link.

If synchronization is completely lost, weight adjustment function **2305** decreases the weight value on the affected path to zero. When synchronization is regained, the weight value for the affected path is gradually increased to its original value. Alternatively, link quality can be measured by evaluating the length of time required for the receiver to acknowledge a synchronization request. In one embodiment, separate transmit and receive tables are used for each transmission path.

When the transmitter receives a SYNC_ACK, the MESS_R is compared with the number of messages transmitted in a window (MESS_T). When the transmitter receives a SYNC_ACK, the traffic probabilities will be examined and adjusted if necessary. MESS_R is compared with the number of messages transmitted in a window (MESS_T). There are two possibilities:

1. If MESS_R is less than a threshold value, THRESH, then the link will be deemed to be unhealthy. If the transmitter was turned off, the transmitter is turned on and the weight P for that link will be set to a minimum value MIN. This will keep a trickle of traffic on the link for monitoring purposes until it recovers. If the transmitter was turned on, the weight P for that link will be set to:

$$P'=\alpha\times MIN+(1-\alpha)\times P \quad (1)$$

Equation 1 will exponentially damp the traffic weight value to MIN during sustained periods of degraded service.

2. If MESS_R for a link is greater than or equal to THRESH, the link will be deemed healthy. If the weight P for that link is greater than or equal to the steady state value S for that link, then P is left unaltered. If the weight P for that link is less than THRESH then P will be set to:

$$P'=\beta\times S+(1-\beta)\times P \quad (2)$$

where β is a parameter such that $0<=\beta<=1$ that determines the damping rate of P.

Equation 2 will increase the traffic weight to S during sustained periods of acceptable service in a damped exponential fashion.

A detailed example will now be provided with reference to FIG. **24**. As shown in FIG. **24**, a first computer **2401** communicates with a second computer **2402** through two routers **2403** and **2404**. Each router is coupled to the other router through three transmission links. As described above, these may be physically diverse links or logical links (including virtual private networks).

Suppose that a first link L1 can sustain a transmission bandwidth of 100 Mb/s and has a window size of 32; link L2 can sustain 75 Mb/s and has a window size of 24; and link L3 can sustain 25 Mb/s and has a window size of 8. The combined links can thus sustain 200 Mb/s. The steady state traffic weights are 0.5 for link L1; 0.375 for link L2, and 0.125 for link L3. MIN=1 Mb/s, THRESH=0.8 MESS_T for each link, α=0.75 and β=0.5. These traffic weights will remain stable until a link stops for synchronization or reports a number of packets received less than its THRESH. Consider the following sequence of events:

1. Link L1 receives a SYNC_ACK containing a MESS_R of 24, indicating that only 75% of the MESS_T (32) messages transmitted in the last window were successfully received. Link 1 would be below THRESH (0.8). Consequently, link L1's traffic weight value would be reduced to 0.12825, while link L2's traffic weight value would be increased to 0.65812 and link L3's traffic weight value would be increased to 0.217938.
2. Link L2 and L3 remained healthy and link L1 stopped to synchronize. Then link L1's traffic weight value would be set to 0, link L2's traffic weight value would be set to 0.75, and link L33's traffic weight value would be set to 0.25.
3. Link L1 finally received a SYNC_ACK containing a MESS_R of 0 indicating that none of the MESS_T (32) messages transmitted in the last window were successfully received. Link L1 would be below THRESH. Link L1's traffic weight value would be increased to 0.005, link L2's traffic weight value would be decreased to 0.74625, and link L3's traffic weight value would be decreased to 0.24875.
4. Link L1 received a SYNC_ACK containing a MESS_R of 32 indicating that 100% of the MESS_T

(32) messages transmitted in the last window were successfully received. Link **L1** would be above THRESH. Link **L1**'s traffic weight value would be increased to 0.2525, while link **L2**'s traffic weight value would be decreased to 0.560625 and link **L3**'s traffic weight value would be decreased to 0.186875.

5. Link **L1** received a SYNC_ACK containing a MESS_R of 32 indicating that 100% of the MESS_T (32) messages transmitted in the last window were successfully received. Link **L1** would be above THRESH. Link **L1**'s traffic weight value would be increased to 0.37625; link **L2**'s traffic weight value would be decreased to 0.4678125, and link **L3**'s traffic weight value would be decreased to 0.1559375.
6. Link **L1** remains healthy and the traffic probabilities approach their steady state traffic probabilities.

B. Use of a DNS Proxy to Transparently Create Virtual Private Networks

A second improvement concerns the automatic creation of a virtual private network (VPN) in response to a domain-name server look-up function.

Conventional Domain Name Servers (DNSs) provide a look-up function that returns the IP address of a requested computer or host. For example, when a computer user types in the web name "Yahoo.com," the user's web browser transmits a request to a DNS, which converts the name into a four-part IP address that is returned to the user's browser and then used by the browser to contact the destination web site.

This conventional scheme is shown in FIG. **25**. A user's computer **2501** includes a client application **2504** (for example, a web browser) and an IP protocol stack **2505**. When the user enters the name of a destination host, a request DNS REQ is made (through IP protocol stack **2505**) to a DNS **2502** to look up the IP address associated with the name. The DNS returns the IP address DNS RESP to client application **2504**, which is then able to use the IP address to communicate with the host **2503** through separate transactions such as PAGE REQ and PAGE RESP.

In the conventional architecture shown in FIG. **25**, nefarious listeners on the Internet could intercept the DNS REQ and DNS RESP packets and thus learn what IP addresses the user was contacting. For example, if a user wanted to set up a secure communication path with a web site having the name "Target.com," when the user's browser contacted a DNS to find the IP address for that web site, the true IP address of that web site would be revealed over the Internet as part of the DNS inquiry. This would hamper anonymous communications on the Internet.

One conventional scheme that provides secure virtual private networks over the Internet provides the DNS server with the public keys of the machines that the DNS server has the addresses for. This allows hosts to retrieve automatically the public keys of a host that the host is to communicate with so that the host can set up a VPN without having the user enter the public key of the destination host. One implementation of this standard is presently being developed as part of the FreeS/WAN project(RFC 2535).

The conventional scheme suffers from certain drawbacks. For example, any user can perform a DNS request. Moreover, DNS requests resolve to the same value for all users.

According to certain aspects of the invention, a specialized DNS server traps DNS requests and, if the request is from a special type of user (e.g., one for which secure communication services are defined), the server does not return the true IP address of the target node, but instead automatically sets up a virtual private network between the target node and the user. The VPN is preferably implemented using the IP address "hopping" features of the basic invention described above, such that the true identity of the two nodes cannot be determined even if packets during the communication are intercepted. For DNS requests that are determined to not require secure services (e.g., an unregistered user), the DNS server transparently "passes through" the request to provide a normal look-up function and return the IP address of the target web server, provided that the requesting host has permissions to resolve unsecured sites. Different users who make an identical DNS request could be provided with different results.

FIG. **26** shows a system employing various principles summarized above. A user's computer **2601** includes a conventional client (e.g., a web browser) **2605** and an IP protocol stack **2606** that preferably operates in accordance with an IP hopping function **2607** as outlined above. A modified DNS server **2602** includes a conventional DNS server function **2609** and a DNS proxy **2610**. A gatekeeper server **2603** is interposed between the modified DNS server and a secure target site **2704**. An "unsecure" target site **2611** is also accessible via conventional IP protocols.

According to one embodiment, DNS proxy **2610** intercepts all DNS lookup functions from client **2605** and determines whether access to a secure site has been requested. If access to a secure site has been requested (as determined, for example, by a domain name extension, or by reference to an internal table of such sites), DNS proxy **2610** determines whether the user has sufficient security privileges to access the site. If so, DNS proxy **2610** transmits a message to gatekeeper **2603** requesting that a virtual private network be created between user computer **2601** and secure target site **2604**. In one embodiment, gatekeeper **2603** creates "hop-blocks" to be used by computer **2601** and secure target site **2604** for secure communication. Then, gatekeeper **2603** communicates these to user computer **2601**. Thereafter, DNS proxy **2610** returns to user computer **2601** the resolved address passed to it by the gatekeeper (this address could be different from the actual target computer) **2604**, preferably using a secure administrative VPN. The address that is returned need not be the actual address of the destination computer.

Had the user requested lookup of a non-secure web site such as site **2611**, DNS proxy would merely pass through to conventional DNS server **2609** the look-up request, which would be handled in a conventional manner, returning the IP address of non-secure web site **2611**. If the user had requested lookup of a secure web site but lacked credentials to create such a connection, DNS proxy **2610** would return a "host unknown" error to the user. In this manner, different users requesting access to the same DNS name could be provided with different look-up results.

Gatekeeper **2603** can be implemented on a separate computer (as shown in FIG. **26**) or as a function within modified DNS server **2602**. In general, it is anticipated that gatekeeper **2703** facilitates the allocation and exchange of information needed to communicate securely, such as using "hopped" IP addresses. Secure hosts such as site **2604** are assumed to be equipped with a secure communication function such as an IP hopping function **2608**.

It will be appreciated that the functions of DNS proxy **2610** and DNS server **2609** can be combined into a single server for convenience. Moreover, although element **2602** is shown as combining the functions of two servers, the two servers can be made to operate independently.

FIG. **27** shows steps that can be executed by DNS proxy server **2610** to handle requests for DNS look-up for secure

hosts. In step **2701**, a DNS look-up request is received for a target host. In step **2702**, a check is made to determine whether access to a secure host was requested. If not, then in step **2703** the DNS request is passed to conventional DNS server **2609**, which looks up the IP address of the target site and returns it to the user's application for further processing.

In step **2702**, if access to a secure host was requested, then in step **2704** a further check is made to determine whether the user is authorized to connect to the secure host. Such a check can be made with reference to an internally stored list of authorized IP addresses, or can be made by communicating with gatekeeper **2603** (e.g., over an "administrative" VPN that is secure). It will be appreciated that different levels of security can also be provided for different categories of hosts. For example, some sites may be designated as having a certain security level, and the security level of the user requesting access must match that security level. The user's security level can also be determined by transmitting a request message back to the user's computer requiring that it prove that it has sufficient privileges.

If the user is not authorized to access the secure site, then a "host unknown" message is returned (step **2705**). If the user has sufficient security privileges, then in step **2706** a secure VPN is established between the user's computer and the secure target site. As described above, this is preferably done by allocating a hopping regime that will be carried out between the user's computer and the secure target site, and is preferably performed transparently to the user (i.e., the user need not be involved in creating the secure link). As described in various embodiments of this application, any of various fields can be "hopped" (e.g., IP source/destination addresses; a field in the header; etc.) in order to communicate securely.

Some or all of the security functions can be embedded in gatekeeper **2603**, such that it handles all requests to connect to secure sites. In this embodiment, DNS proxy **2610** communicates with gatekeeper **2603** to determine (preferably over a secure administrative VPN) whether the user has access to a particular web site. Various scenarios for implementing these features are described by way of example below:

Scenario #1: Client has permission to access target computer, and gatekeeper has a rule to make a VPN for the client. In this scenario, the client's DNS request would be received by the DNS proxy server **2610**, which would forward the request to gatekeeper **2603**. The gatekeeper would establish a VPN between the client and the requested target. The gatekeeper would provide the address of the destination to the DNS proxy, which would then return the resolved name as a result. The resolved address can be transmitted back to the client in a secure administrative VPN.

Scenario #2: Client does not have permission to access target computer. In this scenario, the client's DNS request would be received by the DNS proxy server **2610**, which would forward the request to gatekeeper **2603**. The gatekeeper would reject the request, informing DNS proxy server **2610** that it was unable to find the target computer. The DNS proxy **2610** would then return a "host unknown" error message to the client.

Scenario #3: Client has permission to connect using a normal non-VPN link, and the gatekeeper does not have a rule to set up a VPN for the client to the target site. In this scenario, the client's DNS request is received by DNS proxy server **2610**, which would check its rules and determine that no VPN is needed. Gatekeeper **2603** would then inform the DNS proxy server to forward the request to conventional DNS server **2609**, which would resolve the request and return the result to the DNS proxy server and then back to the client.

Scenario #4: Client does not have permission to establish a normal/non-VPN link, and the gatekeeper does not have a rule to make a VPN for the client to the target site. In this scenario, the DNS proxy server would receive the client's DNS request and forward it to gatekeeper **2603**. Gatekeeper **2603** would determine that no special VPN was needed, but that the client is not authorized to communicate with non-VPN members. The gatekeeper would reject the request, causing DNS proxy server **2610** to return an error message to the client.

C. Large Link to Small Link Bandwidth Management

One feature of the basic architecture is the ability to prevent so-called "denial of service" attacks that can occur if a computer hacker floods a known Internet node with packets, thus preventing the node from communicating with other nodes. Because IP addresses or other fields are "hopped" and packets arriving with invalid addresses are quickly discarded, Internet nodes are protected against flooding targeted at a single IP address.

In a system in which a computer is coupled through a link having a limited bandwidth (e.g., an edge router) to a node that can support a much higher-bandwidth link (e.g., an Internet Service Provider), a potential weakness could be exploited by a determined hacker. Referring to FIG. **28**, suppose that a first host computer **2801** is communicating with a second host computer **2804** using the IP address hopping principles described above. The first host computer is coupled through an edge router **2802** to an Internet Service Provider (ISP) **2803** through a low bandwidth link (LOW BW), and is in turn coupled to second host computer **2804** through parts of the Internet through a high bandwidth link (HIGH BW). In this architecture, the ISP is able to support a high bandwidth to the internet, but a much lower bandwidth to the edge router **2802**.

Suppose that a computer hacker is able to transmit a large quantity of dummy packets addressed to first host computer **2801** across high bandwidth link HIGH BW. Normally, host computer **2801** would be able to quickly reject the packets since they would not fall within the acceptance window permitted by the IP address hopping scheme. However, because the packets must travel across low bandwidth link LOW BW, the packets overwhelm the lower bandwidth link before they are received by host computer **2801**. Consequently, the link to host computer **2801** is effectively flooded before the packets can be discarded.

According to one inventive improvement, a "link guard" function **2805** is inserted into the high-bandwidth node (e.g., ISP **2803**) that quickly discards packets destined for a low-bandwidth target node if they are not valid packets. Each packet destined for a low-bandwidth node is cryptographically authenticated to determine whether it belongs to a VPN. If it is not a valid VPN packet, the packet is discarded at the high-bandwidth node. If the packet is authenticated as belonging to a VPN, the packet is passed with high preference. If the packet is a valid non-VPN packet, it is passed with a lower quality of service (e.g., lower priority).

In one embodiment, the ISP distinguishes between VPN and non-VPN packets using the protocol of the packet. In the case of IPSEC [rfc 2401], the packets have IP protocols **420** and **421**. In the case of the TARP VPN, the packets will have an IP protocol that is not yet defined. The ISP's link guard, **2805**, maintains a table of valid VPNs which it uses to validate whether VPN packets are cryptographically valid.

According to one embodiment, packets that do not fall within any hop windows used by nodes on the low-bandwidth link are rejected, or are sent with a lower quality of service. One approach for doing this is to provide a copy of the IP hopping tables used by the low-bandwidth nodes to the high-bandwidth node, such that both the high-bandwidth and low-bandwidth nodes track hopped packets (e.g., the high-bandwidth node moves its hopping window as valid packets are received). In such a scenario, the high-bandwidth node discards packets that do not fall within the hopping window before they are transmitted over the low-bandwidth link. Thus, for example, ISP **2903** maintains a copy **2910** of the receive table used by host computer **2901**. Incoming packets that do not fall within this receive table are discarded. According to a different embodiment, link guard **2805** validates each VPN packet using a keyed hashed message authentication code (HMAC) [rfc 2104].

According to another embodiment, separate VPNs (using, for example, hopblocks) can be established for communicating between the low-bandwidth node and the high-bandwidth node (i.e., packets arriving at the high-bandwidth node are converted into different packets before being transmitted to the low-bandwidth node).

As shown in FIG. **29**, for example, suppose that a first host computer **2900** is communicating with a second host computer **2902** over the Internet, and the path includes a high bandwidth link HIGH BW to an ISP **2901** and a low bandwidth link LOW BW through an edge router **2904**. In accordance with the basic architecture described above, first host computer **2900** and second host computer **2902** would exchange hopblocks (or a hopblock algorithm) and would be able to create matching transmit and receive tables **2905**, **2906**, **2912** and **2913**. Then in accordance with the basic architecture, the two computers would transmit packets having seemingly random IP source and destination addresses, and each would move a corresponding hopping window in its receive table as valid packets were received.

Suppose that a nefarious computer hacker **2903** was able to deduce that packets having a certain range of IP addresses (e.g., addresses 100 to 200 for the sake of simplicity) are being transmitted to ISP **2901**, and that these packets are being forwarded over a low-bandwidth link. Hacker computer **2903** could thus "flood" packets having addresses falling into the range 100 to 200, expecting that they would be forwarded along low bandwidth link LOW BW, thus causing the low bandwidth link to become overwhelmed. The fast packet reject mechanism in first host computer **3000** would be of little use in rejecting these packets, since the low bandwidth link was effectively jammed before the packets could be rejected. In accordance with one aspect of the improvement, however, VPN link guard **2911** would prevent the attack from impacting the performance of VPN traffic because the packets would either be rejected as invalid VPN packets or given a lower quality of service than VPN traffic over the lower bandwidth link. A denial-of-service flood attack could, however, still disrupt non-VPN traffic.

According to one embodiment of the improvement, ISP **2901** maintains a separate VPN with first host computer **2900**, and thus translates packets arriving at the ISP into packets having a different IP header before they are transmitted to host computer **2900**. The cryptographic keys used to authenticate VPN packets at the link guard **2911** and the cryptographic keys used to encrypt and decrypt the VPN packets at host **2902** and host **2901** can be different, so that link guard **2911** does not have access to the private host data; it only has the capability to authenticate those packets.

According to yet a third embodiment, the low-bandwidth node can transmit a special message to the high-bandwidth node instructing it to shut down all transmissions on a particular IP address, such that only hopped packets will pass through to the low-bandwidth node. This embodiment would prevent a hacker from flooding packets using a single IP address. According to yet a fourth embodiment, the high-bandwidth node can be configured to discard packets transmitted to the low-bandwidth node if the transmission rate exceeds a certain predetermined threshold for any given IP address; this would allow hopped packets to go through. In this respect, link guard **2911** can be used to detect that the rate of packets on a given IP address are exceeding a threshold rate; further packets addressed to that same IP address would be dropped or transmitted at a lower priority (e.g., delayed).

D. Traffic Limiter

In a system in which multiple nodes are communicating using "hopping" technology, a treasonous insider could internally flood the system with packets. In order to prevent this possibility, one inventive improvement involves setting up "contracts" between nodes in the system, such that a receiver can impose a bandwidth limitation on each packet sender. One technique for doing this is to delay acceptance of a checkpoint synchronization request from a sender until a certain time period (e.g., one minute) has elapsed. Each receiver can effectively control the rate at which its hopping window moves by delaying "SYNC ACK" responses to "SYNC_REQ" messages.

A simple modification to the checkpoint synchronizer will serve to protect a receiver from accidental or deliberate overload from an internally treasonous client. This modification is based on the observation that a receiver will not update its tables until a SYNC_REQ is received on hopped address CKPT_N. It is a simple matter of deferring the generation of a new CKPT_N until an appropriate interval after previous checkpoints.

Suppose a receiver wished to restrict reception from a transmitter to 100 packets a second, and that checkpoint synchronization messages were triggered every 50 packets. A compliant transmitter would not issue new SYNC_REQ messages more often than every 0.5 seconds. The receiver could delay a non-compliant transmitter from synchronizing by delaying the issuance of CKPT_N for 0.5 second after the last SYNC_REQ was accepted.

In general, if M receivers need to restrict N transmitters issuing new SYNC_REQ messages after every W messages to sending R messages a second in aggregate, each receiver could defer issuing a new CKPT_N until M×N×W/R seconds have elapsed since the last SYNC_REQ has been received and accepted. If the transmitter exceeds this rate between a pair of checkpoints, it will issue the new checkpoint before the receiver is ready to receive it, and the SYNC_REQ will be discarded by the receiver. After this, the transmitter will re-issue the SYNC_REQ every Ti seconds until it receives a SYNC_ACK. The receiver will eventually update CKPT_N and the SYNC_REQ will be acknowledged. If the transmission rate greatly exceeds the allowed rate, the transmitter will stop until it is compliant. If the transmitter exceeds the allowed rate by a little, it will eventually stop after several rounds of delayed synchronization until it is in compliance. Hacking the transmitter's code to not shut off only permits the transmitter to lose the acceptance window. In this case it can recover the window and proceed only after it is compliant again.

Two practical issues should be considered when implementing the above scheme:

1. The receiver rate should be slightly higher than the permitted rate in order to allow for statistical fluctuations in traffic arrival times and non-uniform load balancing.

2. Since a transmitter will rightfully continue to transmit for a period after a SYNC_REQ is transmitted, the algorithm above can artificially reduce the transmitter's bandwidth. If events prevent a compliant transmitter from synchronizing for a period (e.g. the network dropping a SYNC_REQ or a SYNC_ACK) a SYNC_REQ will be accepted later than expected. After this, the transmitter will transmit fewer than expected messages before encountering the next checkpoint. The new checkpoint will not have been activated and the transmitter will have to retransmit the SYNC_REQ. This will appear to the receiver as if the transmitter is not compliant. Therefore, the next checkpoint will be accepted late from the transmitter's perspective. This has the effect of reducing the transmitter's allowed packet rate until the transmitter transmits at a packet rate below the agreed upon rate for a period of time.

To guard against this, the receiver should keep track of the times that the last C SYNC_REQs were received and accepted and use the minimum of M×N×W/R seconds after the last SYNC_REQ has been received and accepted, 2×M×N×W/R seconds after next to the last SYNC_REQ has been received and accepted, C×M×N×W/R seconds after $(C-1)^{th}$ to the last SYNC_REQ has been received, as the time to activate CKPT_N. This prevents the receiver from inappropriately limiting the transmitter's packet rate if at least one out of the last C SYNC_REQs was processed on the first attempt.

FIG. **30** shows a system employing the above-described principles. In FIG. **30**, two computers **3000** and **3001** are assumed to be communicating over a network N in accordance with the "hopping" principles described above (e.g., hopped IP addresses, discriminator values, etc.). For the sake of simplicity, computer **3000** will be referred to as the receiving computer and computer **3001** will be referred to as the transmitting computer, although full duplex operation is of course contemplated. Moreover, although only a single transmitter is shown, multiple transmitters can transmit to receiver **3000**.

As described above, receiving computer **3000** maintains a receive table **3002** including a window W that defines valid IP address pairs that will be accepted when appearing in incoming data packets. Transmitting computer **3001** maintains a transmit table **3003** from which the next IP address pairs will be selected when transmitting a packet to receiving computer **3000**. (For the sake of illustration, window W is also illustrated with reference to transmit table **3003**). As transmitting computer moves through its table, it will eventually generate a SYNC_REQ message as illustrated in function **3010**. This is a request to receiver **3000** to synchronize the receive table **3002**, from which transmitter **3001** expects a response in the form of a CKPT_N (included as part of a SYNC_ACK message). If transmitting computer **3001** transmits more messages than its allotment, it will prematurely generate the SYNC_REQ message. (If it has been altered to remove the SYNC_REQ message generation altogether, it will fall out of synchronization since receiver **3000** will quickly reject packets that fall outside of window W, and the extra packets generated by transmitter **3001** will be discarded).

In accordance with the improvements described above, receiving computer **3000** performs certain steps when a SYNC_REQ message is received, as illustrated in FIG. **30**. In step **3004**, receiving computer **3000** receives the SYNC_REQ message. In step **3005**, a check is made to determine whether the request is a duplicate. If so, it is discarded in step **3006**. In step **3007**, a check is made to determine whether the SYNC_REQ received from transmitter **3001** was received at a rate that exceeds the allowable rate R (i.e., the period between the time of the last SYNC_REQ message). The value R can be a constant, or it can be made to fluctuate as desired. If the rate exceeds R, then in step **3008** the next activation of the next CKPT_N hopping table entry is delayed by W/R seconds after the last SYNC_REQ has been accepted.

Otherwise, if the rate has not been exceeded, then in step **3109** the next CKPT_N value is calculated and inserted into the receiver's hopping table prior to the next SYNC_REQ from the transmitter **3101**. Transmitter **3101** then processes the SYNC_REQ in the normal manner.

E. Signaling Synchronizer

In a system in which a large number of users communicate with a central node using secure hopping technology, a large amount of memory must be set aside for hopping tables and their supporting data structures. For example, if one million subscribers to a web site occasionally communicate with the web site, the site must maintain one million hopping tables, thus using up valuable computer resources, even though only a small percentage of the users may actually be using the system at any one time. A desirable solution would be a system that permits a certain maximum number of simultaneous links to be maintained, but which would "recognize" millions of registered users at any one time. In other words, out of a population of a million registered users, a few thousand at a time could simultaneously communicate with a central server, without requiring that the server maintain one million hopping tables of appreciable size.

One solution is to partition the central node into two nodes: a signaling server that performs session initiation for user log-on and log-off (and requires only minimally sized tables), and a transport server that contains larger hopping tables for the users. The signaling server listens for the millions of known users and performs a fast-packet reject of other (bogus) packets. When a packet is received from a known user, the signaling server activates a virtual private link (VPL) between the user and the transport server, where hopping tables are allocated and maintained. When the user logs onto the signaling server, the user's computer is provided with hop tables for communicating with the transport server, thus activating the VPL. The VPLs can be torn down when they become inactive for a time period, or they can be torn down upon user log-out. Communication with the signaling server to allow user log-on and log-off can be accomplished using a specialized version of the checkpoint scheme described above.

FIG. **31** shows a system employing certain of the above-described principles. In FIG. **31**, a signaling server **3101** and a transport server **3102** communicate over a link. Signaling server **3101** contains a large number of small tables **3106** and **3107** that contain enough information to authenticate a communication request with one or more clients **3103** and **3104**. As described in more detail below, these small tables may advantageously be constructed as a special case of the synchronizing checkpoint tables described previously. Transport server **3102**, which is preferably a separate computer in communication with signaling server **3101**, contains a smaller number of larger hopping tables **3108**, **3109**, and **3110** that can be allocated to create a VPN with one of the client computers.

According to one embodiment, a client that has previously registered with the system (e.g., via a system administration function, a user registration procedure, or some other method) transmits a request for information from a computer (e.g., a web site). In one variation, the request is

made using a "hopped" packet, such that signaling server **3101** will quickly reject invalid packets from unauthorized computers such as hacker computer **3105**. An "administrative" VPN can be established between all of the clients and the signaling server in order to ensure that a hacker cannot flood signaling server **3101** with bogus packets. Details of this scheme are provided below.

Signaling server **3101** receives the request **3111** and uses it to determine that client **3103** is a validly registered user. Next, signaling server **3101** issues a request to transport server **3102** to allocate a hopping table (or hopping algorithm or other regime) for the purpose of creating a VPN with client **3103**. The allocated hopping parameters are returned to signaling server **3101** (path **3113**), which then supplies the hopping parameters to client **3103** via path **3114**, preferably in encrypted form.

Thereafter, client **3103** communicates with transport server **3102** using the normal hopping techniques described above. It will be appreciated that although signaling server **3101** and transport server **3102** are illustrated as being two separate computers, they could of course be combined into a single computer and their functions performed on the single computer. Alternatively, it is possible to partition the functions shown in FIG. **31** differently from as shown without departing from the inventive principles.

One advantage of the above-described architecture is that signaling server **3101** need only maintain a small amount of information on a large number of potential users, yet it retains the capability of quickly rejecting packets from unauthorized users such as hacker computer **3105**. Larger data tables needed to perform the hopping and synchronization functions are instead maintained in a transport server **3102**, and a smaller number of these tables are needed since they are only allocated for "active" links. After a VPN has become inactive for a certain time period (e.g., one hour), the VPN can be automatically torn down by transport server **3102** or signaling server **3101**.

A more detailed description will now be provided regarding how a special case of the checkpoint synchronization feature can be used to implement the signaling scheme described above.

The signaling synchronizer may be required to support many (millions) of standing, low bandwidth connections. It therefore should minimize per-VPL memory usage while providing the security offered by hopping technology. In order to reduce memory usage in the signaling server, the data hopping tables can be completely eliminated and data can be carried as part of the SYNC_REQ message. The table used by the server side (receiver) and client side (transmitter) is shown schematically as element **3106** in FIG. **31**.

The meaning and behaviors of CKPT_N, CKPT_O and CKPT_R remain the same from the previous description, except that CKPT_N can receive a combined data and SYNC_REQ message or a SYNC_REQ message without the data.

The protocol is a straightforward extension of the earlier synchronizer. Assume that a client transmitter is on and the tables are synchronized. The initial tables can be generated "out of band." For example, a client can log into a web server to establish an account over the Internet. The client will receive keys etc encrypted over the Internet. Meanwhile, the server will set up the signaling VPN on the signaling server.

Assuming that a client application wishes to send a packet to the server on the client's standing signaling VPL:

1. The client sends the message marked as a data message on the inner header using the transmitter's CKPT_N address. It turns the transmitter off and starts a timer T1 noting CKPT_O. Messages can be one of three types: DATA, SYNC_REQ and SYNC_ACK. In the normal algorithm, some potential problems can be prevented by identifying each message type as part of the encrypted inner header field. In this algorithm, it is important to distinguish a data packet and a SYNC_REQ in the signaling synchronizer since the data and the SYNC_REQ come in on the same address.
2. When the server receives a data message on its CKPT_N, it verifies the message and passes it up the stack. The message can be verified by checking message type and and other information (i.e user credentials) contained in the inner header. It replaces its CKPT_O with CKPT_N and generates the next CKPT_N. It updates its transmitter side CKPT_R to correspond to the client's receiver side CKPT R and transmits a SYNC ACK containing CKPT_O in its payload.
3. When the client side receiver receives a SYNC_ACK on its CKPT_R with a payload matching its transmitter side CKPT_O and the transmitter is off, the transmitter is turned on and the receiver side CKPT_R is updated. If the SYNC_ACK's payload does not match the transmitter side CKPT_O or the transmitter is on, the SYNC_ACK is simply discarded.
4. T1 expires: If the transmitter is off and the client's transmitter side CKPT_O matches the CKPT_O associated with the timer, it starts timer T1 noting CKPT_O again, and a SYNC_REQ is sent using the transmitter's CKPT_O address. Otherwise, no action is taken.
5. When the server receives a SYNC_REQ on its CKPT_N it replaces its CKPT_O with CKPT_N and generates the next CKPT_N. It updates its transmitter side CKPT_R to correspond to the client's receiver side CKPT_R and transmits a SYNC_ACK containing CKPT_O in its payload.
6. When the server receives a SYNC_REQ on its CKPT_O, it updates its transmitter side CKPT_R to correspond to the client's receiver side CKPT_R and transmits a SYNC_ACK containing CKPT_O in its payload.

FIG. **32** shows message flows to highlight the protocol. Reading from top to bottom, the client sends data to the server using its transmitter side CKPT_N. The client side transmitter is turned off and a retry timer is turned off The transmitter will not transmit messages as long as the transmitter is turned off. The client side transmitter then loads CKPT_N into CKPT_O and updates CKPT N. This message is successfully received and a passed up the stack. It also synchronizes the receiver i.e, the server loads CKPT_N into CKPT_O and generates a new CKPT_N, it generates a new CKPT_R in the server side transmitter and transmits a SYNC_ACK containing the server side receiver's CKPT_O the server. The SYNC_ACK is successfully received at the client. The client side receiver's CKPT_R is updated, the transmitter is turned on and the retry timer is killed. The client side transmitter is ready to transmit a new data message.

Next, the client sends data to the server using its transmitter side CKPT_N. The client side transmitter is turned off and a retry timer is turned off. The transmitter will not transmit messages as long as the transmitter is turned off. The client side transmitter then loads CKPT_N into CKPT_O and updates CKPT_N. This message is lost. The client side timer expires and as a result a SYNC_REQ is transmitted on the client side transmitter's CKPT_O (this will keep happening until the SYNC_ACK has been received at the client). The SYNC_REQ is successfully received at the server. It synchronizes the receiver i.e, the server loads CKPT_N into CKPT_O and generates a new

CKPT_N. it generates an new CKPT_R in the server side transmitter and transmits a SYNC_ACK containing the server side receiver's CKPT_O the server. The SYNC_ACK is successfully received at the client. The client side receiver's CKPT_R is updated, the transmitter is turned off and the retry timer is killed. The client side transmitter is ready to transmit a new data message.

There are numerous other scenarios that follow this flow. For example, the SYNC_ACK could be lost. The transmitter would continue to re-send the SYNC_REQ until the receiver synchronizes and responds.

The above-described procedures allow a client to be authenticated at signaling server **3201** while maintaining the ability of signaling server **3201** to quickly reject invalid packets, such as might be generated by hacker computer **3205**. In various embodiments, the signaling synchronizer is really a derivative of the synchronizer. It provides the same protection as the hopping protocol, and it does so for a large number of low bandwidth connections.

What is claimed is:

1. A method of transparently creating a virtual private network (VPN) between a client computer and a target computer, comprising the steps of:

(1) generating from the client computer a Domain Name Service (DNS) request that requests an IP address corresponding to a domain name associated with the target computer;

(2) determining whether the DNS request transmitted in step (1) is requesting access to a secure web site; and

(3) in response to determining that the DNS request in step (2) is requesting access to a secure target web site, automatically initiating the VPN between the client computer and the target computer.

2. The method of claim **1**, wherein steps (2) and (3) are performed at a DNS server separate from the client computer.

3. The method of claim **1**, further comprising the step of:

(4) in response to determining that the DNS request in step (2) is not requesting access to a secure target web site, resolving the IP address for the domain name and returning the IP address to the client computer.

4. The method of claim **1**, wherein step (3) comprises the step of, prior to automatically initiating the VPN between the client computer and the target computer, determining whether the client computer is authorized to establish a VPN with the target computer and, if not so authorized, returning an error from the DNS request.

5. The method of claim **1**, wherein step (3) comprises the step of, prior to automatically initiating the VPN between the client computer and the target computer, determining whether the client computer is authorized to resolve addresses of non secure target computers and, if not so authorized, returning an error from the DNS request.

6. The method of claim **1**, wherein step (3) comprises the step of establishing the VPN by creating an IP address hopping scheme between the client computer and the target computer.

7. The method of claim **1**, wherein step (3) comprises the step of using a gatekeeper computer that allocates VPN resources for communicating between the client computer and the target computer.

8. The method of claim **1**, wherein step (2) is performed in a DNS proxy server that passes through the request to a DNS server if it is determined in step (3) that access is not being requested to a secure target web site.

9. The method of claim **5**, wherein step (3) comprises the step of transmitting a message to the client computer to determine whether the client computer is authorized to establish the VPN target computer.

10. A system that transparently creates a virtual private network (VPN) between a client computer and a secure target computer, comprising:

a DNS proxy server that receives a request from the client computer to look up an IP address for a domain name, wherein the DNS proxy server returns the IP address for the requested domain name if it is determined that access to a non-secure web site has been requested, and wherein the DNS proxy server generates a request to create the VPN between the client computer and the secure target computer if it is determined that access to a secure web site has been requested; and

a gatekeeper computer that allocates resources for the VPN between the client computer and the secure web computer in response to the request by the DNS proxy server.

11. The system of claim **10**, wherein the gatekeeper computer creates the VPN by establishing an IP address hopping regime that is used to pseudorandomly change IP addresses in packets transmitted between the client computer and the secure target computer.

12. The system of claim **10**, wherein the gatekeeper computer determines whether the client computer has sufficient security privileges to create the VPN and, if the client computer lacks sufficient security privileges, rejecting the request to create the VPN.

13. A method of establishing communication between one of a plurality of client computers and a central computer that maintains a plurality of authentication tables each corresponding to one of the client computers, the method comprising the steps of:

(1) in the central computer, receiving from one of the plurality of client computers a request to establish a connection;

(2) authenticating, with reference to one of the plurality of authentication tables, that the request received in step (1) is from an authorized client;

(3) responsive to a determination that the request is from an authorized client, allocating resources to establish a virtual private link between the client and a second computer; and

(4) communicating between the authorized client and the second computer using the virtual private link.

14. The method of claim **13**, wherein step (4) comprises the step of communicating according to a scheme by which at least one field in a series of data packets is periodically changed according to a known sequence.

15. The method of claim **14**, wherein step (4) comprises the step of comparing an Internet Protocol (IP) address in a header of each data packet to a table of valid IP addresses maintained in a table in the second computer.

16. The method of claim **15**, wherein step (4) comprises the step of comparing the IP address in the header of each data packet to a moving window of valid IP addresses, and rejecting data packets having IP addresses that do not fall within the moving window.

17. The method of claim **13**, wherein step (2) comprises the step of using a checkpoint data structure that maintains synchronization of a periodically changing parameter known by the central computer and the client computer to authenticate the client.

* * * * *

UNITED STATES PATENT AND TRADEMARK OFFICE

CERTIFICATE OF CORRECTION

PATENT NO. : 6,502,135 B1
DATED : December 31, 2002
INVENTOR(S) : Edmund Colby Munger et al.

It is certified that error appears in the above-identified patent and that said Letters Patent is hereby corrected as shown below:

Title page,
Item [56], **References Cited**, OTHER PUBLICATIONS, insert the following:

-- Search Report (dated 8/20/02), International Application No. PCT/US01/04340
Search Report (dated 8/23/02), International Application No. PCT/US01/13260
James E. Bellaire, "New Statement of Rules – Naming Internet Domains", Internet Newsgroup, July 30, 1995, 1 page.
D. Clark, "US Calls for Private Domain-Name System", Computer, IEEE Computer Society, August 1, 1998, pages 22-25.
August Bequai, "Balancing Legal Concerns Over Crime and Security in Cyberspace", Computer & Security, Vol. 17, No. 4, 1998, pages 293-298.
Rich Winkel, "CAQ: Networking With Spooks: The NET & The Control Of Information", Internet Newsgroup, June 21, 1997, 4 pages. --

Column 48,
Line 2, "VPN target computer" has been replaced with -- VPN with the target computer --.

Signed and Sealed this

Ninth Day of September, 2003

JAMES E. ROGAN
Director of the United States Patent and Trademark Office

US006502135C1

(12) **INTER PARTES REEXAMINATION CERTIFICATE** (0271st)

United States Patent
Munger et al.

(10) **Number: US 6,502,135 C1**
(45) **Certificate Issued: Jun. 7, 2011**

(54) **AGILE NETWORK PROTOCOL FOR SECURE COMMUNICATIONS WITH ASSURED SYSTEM AVAILABILITY**

(75) Inventors: **Edmund Colby Munger**, Crownsville, MD (US); **Douglas Charles Schmidt**, Severna Park, MD (US); **Robert Dunham Short, III**, Leesburg, VA (US); **Victor Larson**, Fairfax, VA (US); **Michael Williamson**, South Riding, VA (US)

(73) Assignee: **Virnetx, Inc.**, Scotts Valley Drive, CA (US)

Reexamination Request:
No. 95/001,269, Dec. 8, 2009

Reexamination Certificate for:
Patent No.: **6,502,135**
Issued: **Dec. 31, 2002**
Appl. No.: **09/504,783**
Filed: **Feb. 15, 2000**

Certificate of Correction issued Sep. 9, 2003.

Related U.S. Application Data

(63) Continuation of application No. 09/429,643, filed on Oct. 29, 1999, now Pat. No. 7,010,604.
(60) Provisional application No. 60/106,261, filed on Oct. 30, 1998, and provisional application No. 60/137,704, filed on Jun. 7, 1999.

(51) **Int. Cl.**
G06F 15/173 (2006.01)

(52) **U.S. Cl.** **709/225**; 709/229; 709/245
(58) **Field of Classification Search** 709/225
See application file for complete search history.

(56) **References Cited**

U.S. PATENT DOCUMENTS

2,895,502 A 7/1959 Roper et al.
4,933,846 A 6/1990 Humphrey et al.
4,988,990 A 1/1991 Warrior
5,276,735 A 1/1994 Boebert et al.
5,303,302 A 4/1994 Burrows

(Continued)

FOREIGN PATENT DOCUMENTS

DE 199 24 575 12/1999
EP 0 814 589 12/1997
EP 836306 A1 4/1998
EP 0 838 930 4/1998
EP 0 858 189 8/1998

(Continued)

OTHER PUBLICATIONS

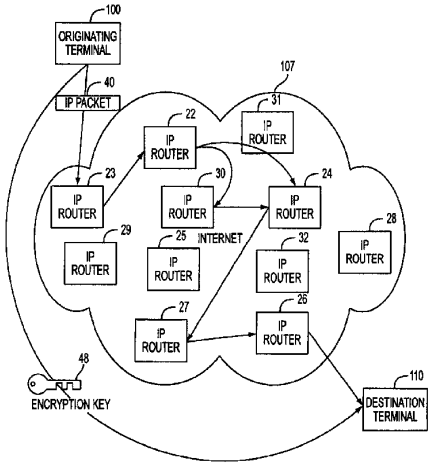
Alan O. Frier et al., "The SSL Protocol Version 3.0", Nov. 18, 1996, printed from http://www.netscape.com/eng/ss13/draft302.txt on Feb. 4, 2002, 56 pages.

(Continued)

Primary Examiner—Andrew L Nalven

(57) **ABSTRACT**

A plurality of computer nodes communicate using seemingly random Internet Protocol source and destination addresses. Data packets matching criteria defined by a moving window of valid addresses are accepted for further processing, while those that do not meet the criteria are quickly rejected. Improvements to the basic design include (1) a load balancer that distributes packets across different transmission paths according to transmission path quality; (2) a DNS proxy server that transparently creates a virtual private network in response to a domain name inquiry; (3) a large-to-small link bandwidth management feature that prevents denial-of-service attacks at system chokepoints; (4) a traffic limiter that regulates incoming packets by limiting the rate at which a transmitter can be synchronized with a receiver; and (5) a signaling synchronizer that allows a large number of nodes to communicate with a central node by partitioning the communication function between two separate entities.



U.S. PATENT DOCUMENTS

5,311,593	A	5/1994	Carmi
5,329,521	A	7/1994	Walsh et al.
5,341,426	A	8/1994	Barney et al.
5,367,643	A	11/1994	Chang et al.
5,384,848	A	1/1995	Kikuchi
5,511,122	A	4/1996	Atkinson
5,559,883	A	9/1996	Williams
5,561,669	A	10/1996	Lenney et al.
5,588,060	A	12/1996	Aziz
5,625,626	A	4/1997	Umekita
5,629,984	A	5/1997	McManis
5,654,695	A	8/1997	Olnowich et al.
5,682,480	A	10/1997	Nakagawa
5,689,566	A	11/1997	Nguyen
5,740,375	A	4/1998	Dunne et al.
5,764,906	A	6/1998	Edelstein et al.
5,771,239	A	6/1998	Moroney et al.
5,774,660	A	6/1998	Brendel et al.
5,787,172	A	7/1998	Arnold
5,796,942	A	8/1998	Esbensen
5,805,801	A	9/1998	Holloway et al.
5,805,803	A	9/1998	Birrell et al.
5,822,434	A	10/1998	Caronni et al.
5,842,040	A	11/1998	Hughes et al.
5,845,091	A	12/1998	Dunne et al.
5,864,666	A	1/1999	Shrader
5,867,650	A	2/1999	Osterman
5,870,610	A	2/1999	Beyda et al.
5,878,231	A	3/1999	Baehr et al.
5,892,903	A	4/1999	Klaus
5,898,830	A	4/1999	Wesinger et al.
5,905,859	A	5/1999	Holloway et al.
5,918,019	A	6/1999	Valencia
5,950,195	A	9/1999	Stockwell et al.
5,996,016	A	11/1999	Thalheimer et al.
6,006,259	A	12/1999	Adelman et al.
6,006,272	A	12/1999	Aravamudan et al.
6,016,318	A	1/2000	Tomoike
6,016,512	A	1/2000	Huitema
6,041,342	A	3/2000	Yamaguchi
6,052,788	A	4/2000	Wesinger et al.
6,055,574	A	4/2000	Smorodinsky et al.
6,061,346	A	5/2000	Nordman
6,061,736	A	5/2000	Rochberger et al.
6,079,020	A	6/2000	Liu
6,081,900	A	6/2000	Subramaniam et al.
6,092,200	A	7/2000	Muniyappa et al.
6,101,182	A	8/2000	Sistanizadeh et al.
6,119,171	A	9/2000	Alkhatib
6,119,234	A	9/2000	Aziz et al.
6,147,976	A	11/2000	Shand et al.
6,157,957	A	12/2000	Berthaud
6,158,011	A	12/2000	Chen et al.
6,168,409	B1	1/2001	Fare
6,173,399	B1	1/2001	Gilbrech
6,175,867	B1	1/2001	Taghadoss
6,178,409	B1	1/2001	Weber et al.
6,178,505	B1	1/2001	Schneider et al.
6,179,102	B1	1/2001	Weber et al.
6,199,112	B1	3/2001	Wilson
6,202,081	B1	3/2001	Naudus
6,222,842	B1	4/2001	Sasyan et al.
6,223,287	B1	4/2001	Douglas et al.
6,226,748	B1	5/2001	Bots et al.
6,226,751	B1	5/2001	Arrow et al.
6,233,618	B1	5/2001	Shannon
6,243,360	B1	6/2001	Basilico
6,243,749	B1	6/2001	Sitaraman et al.
6,243,754	B1	6/2001	Guerin et al.
6,246,670	B1	6/2001	Karlsson et al.
6,256,671	B1	7/2001	Strentzsch et al.
6,262,987	B1	7/2001	Mogul
6,263,445	B1	7/2001	Blumenau
6,286,047	B1	9/2001	Ramanathan et al.
6,298,341	B1	10/2001	Mann et al.
6,301,223	B1	10/2001	Hrastar et al.
6,308,274	B1	10/2001	Swift
6,311,207	B1	10/2001	Mighdoll et al.
6,314,463	B1	11/2001	Abbott et al.
6,324,161	B1	11/2001	Kirch
6,330,562	B1	12/2001	Boden et al.
6,332,158	B1	12/2001	Risley et al.
6,333,272	B1	12/2001	McMillin et al.
6,338,082	B1	1/2002	Schneider
6,353,614	B1	3/2002	Borella et al.
6,430,155	B1	8/2002	Davie et al.
6,430,610	B1	8/2002	Carter
6,487,598	B1	11/2002	Valencia
6,502,135	B1	12/2002	Munger et al.
6,505,232	B1	1/2003	Mighdoll et al.
6,510,154	B1	1/2003	Mayes et al.
6,549,516	B1	4/2003	Albert et al.
6,557,037	B1	4/2003	Provino
6,571,296	B1	5/2003	Dillon
6,571,338	B1	5/2003	Shaio et al.
6,581,166	B1	6/2003	Hirst et al.
6,618,761	B2	9/2003	Munger et al.
6,671,702	B2	12/2003	Kruglikov et al.
6,687,551	B2	2/2004	Steindl
6,687,746	B1	2/2004	Shuster et al.
6,701,437	B1	3/2004	Hoke et al.
6,714,970	B1	3/2004	Fiveash et al.
6,717,949	B1	4/2004	Boden et al.
6,752,166	B2	6/2004	Lull et al.
6,757,740	B1	6/2004	Parekh et al.
6,760,766	B1	7/2004	Sahlqvist
6,826,616	B2	11/2004	Larson et al.
6,839,759	B2	1/2005	Larson et al.
6,937,597	B1	8/2005	Rosenberg et al.
7,010,604	B1	3/2006	Munger et al.
7,039,713	B1	5/2006	Van Gunter et al.
7,072,964	B1	7/2006	Whittle et al.
7,133,930	B2	11/2006	Munger et al.
7,167,904	B1	1/2007	Devarajan et al.
7,188,175	B1	3/2007	McKeeth
7,188,180	B2	3/2007	Larson et al.
7,197,563	B2	3/2007	Sheymov et al.
7,353,841	B2	4/2008	Kono et al.
7,461,334	B1	12/2008	Lu et al.
7,490,151	B2	2/2009	Munger et al.
7,493,403	B2	2/2009	Shull et al.
2001/0049741	A1	12/2001	Skene et al.
2002/0004898	A1	1/2002	Droge
2004/0199493	A1	10/2004	Ruiz et al.
2004/0199520	A1	10/2004	Ruiz et al.
2004/0199608	A1	10/2004	Rechterman et al.
2004/0199620	A1	10/2004	Ruiz et al.
2005/0055306	A1	3/2005	Miller et al.
2007/0208869	A1	9/2007	Adelman et al.
2007/0214284	A1	9/2007	King et al.
2007/0266141	A1	11/2007	Norton
2008/0235507	A1	9/2008	Ishikawa et al.

FOREIGN PATENT DOCUMENTS

GB	2 317 792	4/1998
GB	2 334 181 A	8/1999
JP	62-214744	9/1987
JP	04-363941	12/1992
JP	09-018492	1/1997
JP	10-070531	3/1998
WO	WO 9827783 A	6/1998

WO	WO 98/27783	6/1998
WO	WO 98 55930	12/1998
WO	WO 98 59470	12/1998
WO	WO 99 38081	7/1999
WO	WO 99 48303	9/1999
WO	WO 00/17775	3/2000
WO	WO 001/17775	3/2000
WO	WO 00/70458	11/2000
WO	WO 01/016766	3/2001
WO	WO 01 50688	7/2001

OTHER PUBLICATIONS

August Bequai, "Balancing Legal Concerns Over Crime and Security in Cyberspace", Computer & Security, vol. 17, No. 4, 1998, pp. 293–298.

D. B. Chapman et al., "Building Internet Firewalls", Nov. 1995, pp. 278–375.

D. Clark, "US Calls for Private Domain–Name System", Computer, IEEE Computer Society, Aug. 1, 1998, pp. 22–25.

Davila J et al, "Implementation of Virtual Private Networks at the Transport Layer", Information Security, Second International Work–shop, ISW'99. Proceedings (Lecture Springer–Verlag Berlin, Germany, [Online] 1999, pp. 85–102, XP002399276, ISBN 3–540–666.

Dolev, Shlomi and Ostrovsky, Rafil, "Efficient Anonymous Multicast and Reception" (Extended Abstact), 16 pages.

Donald E. Eastlake, 3rd, "Domain Name System Security Extensions", Internet Draft, Apr. 1998, pp. 1–51.

F. Halsall, "Data Communications, Computer Networks and Open Systems", Chapter 4, Protocol Basics, 1996, pp. 198–203.

Fasbender, Kesdogan, and Kubitz: "Variable and Scalable Security" Protection of Location Information in Mobile IP, IEEE publication, 1996, pp. 963–967.

Glossary for the Linux FreeS/WAN project, printed from http://liberty.freeswan.org/freeswan_trees/freeswan–1.3/doc/glossary.html on Feb. 21, 2002, 25 pages.

J. Gilmore, "Swan: Securing the Internet against Wiretapping", printed from http://liberty.freeswan.org/freeswan_trees/freeswan–1.3/doc/rationale.html on Feb. 21, 2002, 4 pages.

James E. Bellaire, "New Statement of Rules–Naming Internet Domains", Internet Newsgroup, Jul. 30, 1995, 1 page.

Jim Jones et al., "Distributed Denial of Service Attacks: Defenses", Global Integrity Corporation. 2000, pp. 1–14.

Laurie Wells (LANCASTERBIBELMAIL MSN COM); "Subject: Security Icon" USENET Newsgroup, Oct. 19, 1998, XP002200606, 1 page.

Linux FreeS/WAN Index File, printed from http://liberty.freewan.org/freeswan_trees/freeswan–1.3/doc/ on Feb. 21, 2002, 3 Pages.

P. Srisuresh et al., "DNS extensions to Network address Translators (DNS_ALG)", Internet Draft, Jul. 1998, pp. 1–27.

RFC 2401 (dated Nov. 1998) Security Architecture for the Internet Protocol (RTP).

RFC 2543–SIP (dated Mar. 1999): Session Initiation Protocol (SIP or SIPS).

Rich Winkel, "CAQ: Networking With Spooks: The NET & The Control Of information", Internet Newsgroup, Jun. 21, 1997, 4 pages.

Rubin, Aviel D., Geer, Daniel, and Ranum, Marcus J. (Wiley Computer Publishing), "Web Security Sourcebook", pp. 82–94.

Search Report (dated Aug. 20, 2002), International Application No. PCT/US01/04340.

Search Report (dated Aug. 23, 2002), International Application No. PCT/US01/13260.

Search Report (dated Oct. 7, 2002), International Application No. PCT/US01/13261.

Search Report, IPER (dataed Nov. 13, 2002), International Application No. PCT/US01/04340.

Search Report, IPER (dated Feb.6, 2002), International Application No. PCT/US01/13261.

Search Report, IPER (dated Jan. 14, 2003), International Application No. PCT/US01/13260.

Sankar, A.U. "A verified sliding window protocol with variable flow control". Proceedings of ACM SIGCOMM conferece on Communications architectures & protocols. pp. 84–91, ACM Press, NY, NY 1986.

Shree Murthy et al., "Congestion–Oriented Shortest Multi–path Routing", Proceedings of IEEE INFOCOM, 1996, pp. 1028–1036.

W. Stallings, "Cryptography And Network Security", 2nd, Edition, Chapter 13, IP Security, Jun. 8, 1998, pp. 399–440.

Fasbender, A. et al., Variable and Scalable Security: Protection of Location Information in Mobile IP, IEEE VTS, 46th, 1996, 5 pp.

156. Finding Your Way Through the VPN Maze (1999) ("PGP").

WatchGuard Technologies, Inc., WatchGuard LiveSecurity for MSS Powerpoint (Feb. 14 2000) (resubmitted).

WatchGuard Technologies, Inc., MSS Version 2.5, Add–On for WatchGuard SOHO Release Notes (Jul. 21, 2000).

Yuan Dong Feng, "A novel scheme combining interleaving technique with cipher in Rayleigh fading channels," Proceedings of the International Conference on Communication technology, 2:S47–02–1–S47–02–4 (1998).

D.W. Davies and W.L. Price, edited by Tadahiro Uezona, "Network Security", Japan, Nikkei McGraw–Hill, Dec. 5, 1958, First Edition, first copy, p. 102–108.

U.S. Appl. No. 60/134,547 filed May 17, 1999, Victor Sheymov.

U.S. Appl. No. 60/151,563 filed Aug. 31, 1999, Bryan Whittles.

U.S. Appl. No. 09/399,753 filed Sep. 22, 1998, Graig Miller et al.

Microsoft Corporation's Fourth Amended Invalidity Contentions dated Jan. 5, 2009, *VirnetX Inc. and Science Applications International Corp.*v. *Microsoft Corporation*.

Appendix A of the Microsoft Corporation's Fourth Amended Invalidity Contentions dated Jan. 5, 2009.

Concordance Table For the References Cited in Tables on pp. 6–15, 71–80 and 116–124 of the Microsoft Corporation's Fourth Amended Invalidity Contentions dated Jan. 5, 2009.

I. P. Mockapetris, "DNS Encoding of Network Names and Other Types," Network Working Group, RFC 1101 (Apr. 1989) (RFC1101, DNS SRV).

DNS–related corresponding dated Sep. 7, 1993 to Sep. 20, 1993. (Pre KX, KX Records).

R. Atkinson, "An Internetwork Authentication Architecture," Naval Research Laboratory, Center for High Assurance Computing Systems (Aug. 5, 1993). (Atkinson NRL, KX Records).

Henning Schulzrinne, *Personal Mobility For Multimedia Services In The Internet,* Proceedings of the Interactive Distributed Multimedia Systems and Services European Workshop at 143 (1996) (Schulzrinne 96).

Microsoft Corp., *Microsoft Virtual Private Networking: Using Point–to–Point Tunneling Protocol for Low–Cost, Secure, Remote Access Across the Internet* (1996) (printed from 1998 PDC DVD–ROM) (Point to Point, Microsoft Prior Art VPN Technology).

"Safe Surfing: How to Build a Secure World Wide Web Connection," IBM Technical Support Organization, (Mar. 1996). (Safe Surfing, Website Art).

Goldschlag, et al., "Hiding Routing Information," Workshop on Information Hiding, Cambridge, UK (May 1996). (Goldschlag II, Onion Routing).

"IPSec Minutes From Montreal", IPSEC Working Group Meeting Notes, http://www.sandleman.ca/ipsec/1996/08/msg00018.html (Jun. 1996). (IPSec Minutes, FreeS/WAN).

J.M. Galvin, "Public Key Distribution with Secure DNS," Proceedings of the Sixth USENIX UNIX Security Symposium, San Jose, California, Jul. 1996. (Galvin, DNSSEC).

J. Gilmore, et al. "Re: Key Management, anyone? (DNS Keying)," IPSec Working Group Mailing List Archives (Aug. 1996). (Gilmore DNS, FreeS/WAN).

H. Orman, et al."Re: Re: DNS? was Re: Key Management, anyone?" IETF IPSec Working Group Mailing List Archive (Aug. 1996/Sep. 1996). (Orman DNS, FreeS/WAN).

Arnt Gulbrandsen & Paul Vixie, *A DNS RR for specifying the location of services* (*DNS SRV*), IETF RFC 2052 (Oct. 1996). (RFC 2052, DNS SRV).

Freier, et al. "The SSL Protocol Version 3.0," Transport Layer Security Working Group (Nov. 18, 1996). (SSL, Underlying Security Technology).

M. Handley, H. Schulzrinne, E. Schooler, Internet Engineering Task Force, Internet Draft, (Dec. 2, 1996). (RFC 2543 Internet Draft 1).

M.G. Reed, et al. "Proxies for Anonymous Routing," 12th Annual Computer Security Applications Conference, San Diego, CA, Dec. 9–13, 1996. (Reed, Onion Routing).

Kenneth F. Alden & Edward P. Wobber, *The AltaVista Tunnel: Using the Internet to Extend Corporate Networks,* Digital Technical Journal (1997) (Alden, AltaVista.

Automative Industry Action Group, "ANX Release 1 Document Publication," AIAG (1997). (AIAG, ANX).

Automative Industry Action Group, "ANX Release 1 Draft Document Publication," AIAG Publications (1997). (AIAG Release, ANX).

Aventail Corp., "AutoSOCKS v. 2.1 Datasheet," available at http://www.archive.org/web/19970212013409/www.aventail.com/prod/autosk2ds.html (1997). (AutoSOCKS, Aventail).

Aventail Corp. "Aventail VPN Data Sheet," available at http://www.archive.org/web/19970212013043/www.aventail.com/prod/vpndata.html (1997). (Data Sheet, Aventail).

Aventail Corp., "Directed VPN Vs. Tunnel," available at http://web.archive.org/web/19970620030312/www.aventail.com/educate/directvpn.html (1997). (Directed VPN, Aventail).

Aventail Corp., "Managing Corporate Access to the Internet," Aventail AutoSOCKS White Paper available at http://web.archive.org/web/19970620030312/www.aventail.com/educate/whitepaper/ipmwp.html (1997). (Corporate Access, Aventail).

Aventail Corp., "Socks Version 5," Aventail Whitepaper, available at http://web.archive.org/web/19970620030312/www.aventail.com/educate/whitepaper/sockswp.html (1997). (Socks, Aventail).

Aventail Corp., "VPN Server V2.0 Administration Guide," (1997). (VPN, Aventail).

Goldschlag, et al. "*Privacy on the Internet,*" Naval Research Laboratory, Center for High Assurance Computer Systems (1997). (Goldschlag I, Onion Routing).

Microsoft Corp., *Installing Configuring and Using PPTP with Microsoft Clients and Servers* (1997). (Using PPTP, Microsoft Prior Art VPN Technology).

Microsoft Corp., *IP Security for Microsoft Windows NT Server 5.0* (1997) (printed from 1998 PDC DVD–ROM). (IP Security, Microsoft Prior Art VPN Technology).

Microsoft Corp., *Microsoft Windows NT Active Directory: An Introduction to the Next Generation Directory Services* (1997) (printed from 1998 PDC DVD–ROM). (Directory, Microsoft Prior Art VPN Technology).

Microsoft Corp. *Routing and Remote Access Service for Windows NT Server NewOpportunities Today and Looking Ahead* (1997) (printed from 1998 PDC DVD–ROM). (Routing, Microsoft Prior Art VPN Technology).

Microsoft Corp., *Understanding Point–to–Point Tunneling Protocol PPTP* (1997) (printed from 1998 PDC DVD–ROM). (Understanding PPTP, Microsoft Prior Art VPN Technology).

J. Mark Smith et al., *Protecting a Private Network: The AltaVista Firewall,* Digital Technical Journal (1997). (Smith, AltaVista).

Naganand Doraswamy *Implementation of Virtual Private Networks* (*VPNs*) *with IPSecurity*, <draft–ietf–ipsec–vpn–00.txt> (Mar. 12, 1997). (Doraswamy).

M. Handley, H. Schulzrinne, E. Schooler, Internet Engineering Task Force, Internet Draft, (Mar. 27, 1997). (RFC 2543 Internet Draft 2).

Aventail Corp., "Aventail and Cybersafe to Provide Secure Authentication For Internet and Intranet Communication," Press Release, Apr. 3, 1997. (Secure Authentication, Aventail).

D. Wagner, et al. "Analysis of the SSL 3.0 Protocol," (Apr. 15, 1997). (Analysis, Underlying Security Technologies).

Automotive Industry Action Group, "ANXO Certification Authority Service and Directory Service Definition for ANX Release 1," AIAG Telecommunications Project Team and Bellcore (May 9, 1997). (AIAG Definition, ANX).

Automotive Industry Action Group, "ANXO Certification Process and ANX Registration Process Definition for ANX Release 1," AIAG Telecommunications Project Team and Bellcore (May 9, 1997). (AIAG Certification, ANX).

Aventail Corp. "Aventail Announces the First VPN Solution to Assure Interoperability Across Emerging Security Protocols," Jun. 2, 1997. (First VPN, Aventail).

Syverson, et al. "Private Web Browsing," Naval Research Laboratory, Center for High 8 Assurance Computer Systems (Jun. 2, 1997). (Syverson, Onion Routing).

Bellcore, "Metrics, Criteria, and Measurement Technique Requirements for ANX Release 1," AIAG Telecommunications Project Team and Bellcore (Jun. 16, 1997). (AIAG Requirements, ANX).

M. Handley, H. Schulzrinne, E. Schooler, Internet Engineering Task Force, Internet Draft, (Jul. 31, 1997). (RFC 2543 Internet Draft 3).

R. Atkinson, "Key Exchange Delegation Record for the DNS," Network Working Group, RFC 2230 (Nov. 1997). (RFC 2230, KX Records).

M. Handley, H. Schulzrinne, E. Schooler, Internet Engineering Task Force, Internet Draft, (Nov. 11, 1997). (RFC 2543 Internet Draft 4).

1998 Microsoft Professional Developers Conference DVD ("1998 PDC DVD–ROM") (including screenshots captured therefrom and produced as MSFTVX 00018827–00018832). (Conference, Microsoft Prior Art VPN Technology).

Microsoft Corp., *Virtual Private Networking An Overview* (1998) (printed from 1998 PDC DVD–ROM) (Overview, Microsoft Prior Art VPN Technology).

Microsoft Corp., *Windows NT 5.0 Beta Has Public Premiere at Seattle Mini–Camp Seminar attendees get first look at the performance and capabilities of Windows NT 5.0* (1998) (available at hap //www.microsoft.com/presspass/features/1998/10–19nt5.mspxpftrue). (NT Beta, Microsoft Prior Art VPN Technology).

"What ports does SSL use" available at stason.org/TULARC/security/ssl–talk/3–4–What–ports–does–ssl–use.html (1998). (Ports, DNS SRV).

Aventail Corp., "Aventail VPN V2.6 Includes Support for More Than Ten Authentication Methods Making Extranet VPN Development Secure and Simple," Press Release, Jan. 19, 1998. (VPN V2.6, Aventail).

R. G. Moskowitz, "Network Address Translation Issues with IPsec," Internet Draft, Internet Engineering Task Force, Feb. 6, 1998. (Moskowitz).

H. Schulzrinne, et al, "Internet Telephony Gateway Location," Proceedings of IEEE INfocom '98, The Conference on Computer Communications, vol. 2 (Mar. 29–Apr. 2, 1998). (Gateway, Schulzrinne).

C. Huitema, 45 al. "Simple Gateway Control Protocol," Version 1.0 (May 5, 1998). (SGCP).

DISA "Secret Internet Protocol Router Network," SIPRNET Program Management Office (D3113) DISN Networks, DISN Transmission Services (May 8, 1998). (DISA, SIPRNET).

M. Handley, H. Schulzrinne, E. Schooler, Internet Engineering Task Force, Internet Draft, (May 14, 1998). (RFC 2543 Internet Draft 5).

M. Handley, H. Schulzrinne, E. Schooler, Internet Engineering Task Force, Internet Draft, (Jun. 17, 1998). (RFC 2543 Internet Draft 6).

D. McDonald, et al. "PF_KEY Management API, Version 2," Network Working Group, RFC 2367 (Jul. 1998). (RFC 2367).

M. Handley, H. Schulzrinne, E. Schooler, Internet Engineering Task Force, Internet Draft, (Jul. 16, 1998). (RFC 2543 Internet Draft 7).

M. Handley, H. Schulzrinne, E. Schooler, Internet Engineering Task Force, Internet Draft, (Aug. 7, 1998). (RFC 2543 Internet Draft 8).

Microsoft Corp., *Company Focuses on Quality and Customer Feedback* (Aug. 18, 1998). (Focus, Microsoft Prior Art VPN Technology).

M. Handley, H. Schulzrinne, E. Schooler, Internet Engineering Task Force, Internet Draft, (Sep 18, 1998). (RFC 2543 Internet Draft 9).

Atkinson, et al. "Security Architecture for the Internet Protocol," Network Working Group, RFC 2401 (Nov. 1998). (RFC 2401, Underlying Security Technologies).

M. Handley, H. Schulzrinne, E. Schooler, Internet Engineering Task Force, Internet Draft, (Nov. 12, 1998). (RFC 2543 Internet Draft 10) 9.

Donald Eastlake, *Domain Name System Security Extensions,* IETF–DNS Security Working Group (Dec. 1998). (DNS-SEC–7).

M. Handley, H. Schulzrinne, E. Schooler, Internet Engineering Task Force, Internet Draft, (Dec. 15, 1998). (RFC 2543 Internet Draft 11).

Aventail Corp., "Aventail Connect 3.1/2.6Administrator's Guide," (1999). (Aventail Administrator 3.1, Aventail).

Aventail Corp., "Aventail Connect 3.1/2.6 User's Guide," (1999). (Aventail Administrator 3.1, Aventail).

Aventail Corp., "Aventail ExtraWeb Server v3.2 Administrator's Guide," (1999). (Aventail ExtraWeb 3.2, Aventail).

Kaufman et al, "Implementing IPsec," (Copyright 1999). (Implementing IPSEC, VPN References).

Network Solutions, Inc. "Enabling SSL," NSI Registry (1999). (Enabling SSL, Underlying Security Technologies).

Check Point Software Technologies Ltd. (1999) (Check Point, Checkpoint FW).

Arnt Gulbrandsen & Paul Vixie, *A DNS RR for specifying the location of services* (*DNS SRV*), <draft–ietf–dnsind–frc2052bis–02.txt> (Jan. 1999). (Gulbrandsen 99, DNS SRV).

C. Scott, et al. *Virtual Private Networks,* O'Reilly and Associates, Inc., 2nd ed. (Jan. 1999). (Scott VPNs).

M. Handley, H. Schulzrinne, E. Schooler, Internet Engineering Task Force, Internet Draft, (Jan. 15, 1999). (RFC 2543 Internet Draft 12).

Goldschlag, et al., "Onion Routing for Anonymous and Private Internet Connections," Naval Research Laboratory, Center for High Assurance Computer Systems (Jan. 28, 1999). (Goldschlag III, Onion Routing).

H. Schulzrinne, "Internet Telephony: architecture and protocols—an IETF perspective," Computer Networks, vol. 31, No. 3 (Feb. 1999). (Telephony, Schulzrinne).

M. Handley, et al, "SIP: Session Initiation Protocol," Network Working Group, RFC 2543 and Internet Drafts (Dec. 1996–Mar. 1999). (Handley, RFC 2543).

FreeS/WAN Project, *Linux FreeS/WAN Compatibility Guide* (Mar. 4, 1999). (FreeS/WAN Compatibility Guide, FreeS/WAN).

Telcordia Technologies, "ANX Release 1 Document Corrections," AIAG (May 11, 1999). (Telcordia, ANX).

Ken Hornstein & Jeffrey Altman, *Distributing Kerberos KDC and Realm Information with DNS* <draft–eitf–cat–krb–dns–locate–oo.txt> (Jun. 21, 1999). (Hornstein, DNS SRV).

Bhattacharya et al. "An LDAP Schema for Configuration and Administration of IPSec Based Virtual Private Networks (VPNs)", IETF Internet Draft (Oct. 1999). (Bhattcharya LDAP VPN).

B. Patel, et al, "DHCP Configuration of IPSEC Tunnel Mode," IPSEC Working Group, Internet Draft 02 (Oct. 15, 1999). (Patel).

Goncalves, et al. Check Point FireWall—1 Administration Guide, McGraw–Hill Companies (2000). (Goncalves, Checkpoint FW).

"Building a Microsoft VPN: A Comprehensive Collection of Microsoft Resources," FirstVPN, (Jan. 2000). (FirstVPN Microsoft).

Gulbrandsen, Vixie & Esibov, *A DNS RR for specifying the location of services (DNS SRV)*, IETF RFC 2782 (Feb. 2000). (RFC 2782, DNS SRV).

Mitre Organization, "Technical Description," Collaborative Operations in Joint Expeditionary Force Experiment (JEFX) 99 (Feb. 2000). (Mitre, SIPRNET).

H. Schulzrinne, et al. "Application–Layer Mobility Using SIP," Mobile Computing and Communications Review, vol. 4, No. 3, pp. 47–57 (Jul. 2000). (Application, SIP).

Kindred et al, "Dynamic VPN Communities: Implementation and Experience," DARPA Information Survivability Conference and Exposition II (Jun. 2001). (DARPA, VPN Systems).

ANX 101: Basic ANX Service Outline. (Outline, ANX).

ANX 201: Advanced ANX Service. (Advanced, ANX).

Appendix A: Certificate Profile for ANX IPsec Certificates. (Appendix, ANX).

Assured Digital Products. (Assured Digital).

Aventail Corp., "Aventail AutoSOCKS the Client Key to Network Security," Aventail Corporation White Paper. (Network Security, Aventail).

Cindy Moran, "DISN Data Networks: Secret Internet Protocol Router Network (SIPRNet)." (Moran, SIPRNET).

Data Fellows F–Secure VPN+ (F–Secure VPN+).

Interim Operational Systems Doctrine for the Remote Access Security Program (RASP) Secret Dial–In Solution. (RASP, SIPRNET).

Onion Routing, "Investigation of Route Selection Algorithms," available at http://www.onion–router.net/Archives/Route/Index.html. (Route Selection, Onion Routing).

Secure Computing, "Buttet–Proofing an Army Net," Washington Technology. (Secure, SIPRNET).

Sparta "Dynamic Virtual Private Network," (Sparta, VPN Systems).

Standard Operation Procedure for Using the 1910 Secure Modems. (Standard, SIPRNET).

Publically available emails relating to FreeS/WAN (MSFTVX00018833–MSFTVX00019206). (FreeS/WAN emails, FreeS/WAN).

Kaufman et al., "implementing IPsec," (Copyright 1999) (Implementing IPsec).

Network Associates *Gauntlet Firewall For Unix User's Guide Version 5.0* (1999). (Gauntlet User's Guide—Unix, Firewall Products).

Network Associates *Gauntlet Firewall For Windows NT Getting Started Guide Version 5.0* (1999) (Gauntlet Getting Started Guide—NT, Firewall Products).

Network Associates *Gauntlet Firewall For Unix Getting Started Guide Version 5.0* (1999) (Gauntlet Unix Getting Started Guide, Firewall Products).

Network Associates *Release Notes Gauntlet Firewall for Unix 5.0* (Mar. 19, 1999) (Gauntlet Unix Release Notes, Firewall Products).

Network Associates *Gauntlet Firewall For Windows NT Administrator's Guide Version 5.0* (1999) (Gauntlet NT Administrator's Guide, Firewall Products).

Trusted Information Systems, Inc. *Gauntlet Internet Firewall Firewall–to–Firewall Encryption Guide Version 3.1* (1996) (Gauntlet Firewall–to–Firewall, Firewall Products).

Network Associates *Gauntlet Firewall Global Virtual Private Network User's Guide for Windows NT Version 5.0* (1999) (Gauntlet NT GVPN, GVPN).

Network Associates *Gauntlet Firewall For Unix Global Virtual Private Network User's Guide Version 5.0* (1999) (Gauntlet Unix GVPN, GVPN)

Dan Sterne *Dynamic Virtual Private Networks* (May 23, 2000) (Sterne DVPN, DVPN).

Darrell Kindred *Dynamic Virtual Private Networks (DVPN)* (Dec. 21, 1999) (Kindred DVPN, DVPN).

Dan Sterne et al. *TIS Dynamic Security Perimeter Research Project Demonstration* (Mar. 9, 1998) (Dynamic Security Perimeter, DVPN).

Darrell Kindred *Dynamic Virtual Private Networks Capability Description* (Jan. 5, 2000) (Kindred DVPN Capability, DVPN) 11.

Oct. 7, and 28 1997 email from Domenic J. Turchi Jr. (SPARTA00001712–1714, 1808–1811) (Turchi DVPN email, DVPN).

James Just & Dan Sterne *Security Quickstart Task Update* (Feb. 5, 1997) (Security Quickstart, DVPN).

Virtual Private Network Demonstration dated Mar. 21, 1998 (SPARTA00001844–54) (DVPN Demonstration, DVPN).

GTE Internetworking & BBN Technologies *DARPA Information Assurance Program Integrated Feasibility Demonstration* (IFD) 1.1 Plan (Mar. 10, 1998) (IFD 1.1, DVPN).

Microsoft Corp. Windows NT Server Product Documentation: Administration Guide—Connection Point Services, available at http://www.microsoft.com/technet/archive/winntas/proddocs/inetconctservice/cpsops.mspx (Connection Point Services) (Although undated, this reference refers to the operation of prior art versions of Microsoft Windows. Accordingly, upon information and belief, this reference is prior art to the patents–insuit).

Microsoft Corp. Windows NT Servier Product Documentation: Administration Kit Guide—Connection Manager, available at http://www.microsoft.com/technet/archive/winntas/proddocs/inetconctservice/cmak.mspx (Connection Manager) (Although undated, this reference refers to the operation of prior art versions of Microsoft Windows such as Windows NT 4.0. Accordingly, upon information and belief, this reference is prior art to the patents–in–suit.).

Microsoft Corp. Autodial Heuristics, available at http://support.microsoft.com/kb/164249 (Autodial Heuristics) (Although undated, this reference refers to the operation of prior art versions of Microsoft Windows such as Windows NT 4.0. Accordingly, upon information and belief, this reference is prior art to the patents–in–suit.).

Microsoft Corp., Cariplo: Distributed Component Object Model, (1996) available at http://msdn2.microsoft.com/en–us/library/ms809332(printer).aspx (Cariplo I).

Marc Levy, COM Internet Services (Apr. 23, 1999), available at http://msdn2.microsoft.com/en–us/library/ms809302(printer).aspx (Levy).

Markus Horstmann and Mary Kirtland, DCOM Architecture (Jul. 23, 1997), available at http://msdn2.microsoft.com/en–us/library/ms809311(printer).aspx (Horstmann).

Microsoft Corp., DCOM: A Business Overview (Apr. 1997), available at http://msdn2.microsoft.com/en–us/library/ms809320(printer).aspx (DCOM Business Overview I).

Microsoft Corp., DCOM Technical Overview (Nov. 1996), available at http://msdn2.microsoft.com/en–us/library/ms809340(printer).aspx (DCOM Technical Overview I).

Microsoft Corp., DCOM Architecture White Paper (1998) available in PDC DVD–ROM (DCOM Architecture).

Microsoft Corp, DCOM—The Distributed Component Object Model, A Business Overview White Paper (Microsoft 1997) available in PDC DVD–ROM (DCOM Business Overview II).

Microsoft Corp., DCOM—Cariplo Home Banking Over The Internet White Paper (Microsoft 1996) available in PDC DVD–ROM (Cariplo II).

Microsoft Corp., DCOM Solutions in Action White Paper (Microsoft 1996) available in PDC DVD–ROM (DCOM Solutions in Action).

Microsoft Corp., DCOM Technical Overview White Paper (Microsoft 1996) available 12 in PDC DVD–ROM (DCOM Technical Overview II).

125. Scott Suhy & Glenn Wood, DNS and Microsoft Windows NT 4.0 (1996) available at http://msdn2.microsoft.com/en–us/library/ms810277(printer).aspx (Suhy).

126. Aaron Skonnard, *Essential Winlnet* 313–423 (Addison Wesley Longman 1998) (Essential Winlnet).

Microsoft Corp. Installing, Configuring, and Using PPTP with Microsoft Clients and Servers, (1998) available at http://msdn2.microsoft.com/enus/library/ms811078 (printer).aspx (Using PPTP).

Microsoft Corp. Internet Connection Services for MS RAS, Standard Edition, http://www.microsoft.com/technet/archive/winntas/proddocs/inetconctservice/bcgstart.mspx (Internet Connection Services I).

Microsoft Corp. Internet Connection Services for RAS, Commercial Edition, available athttp://www.microsoft.com/technet/archive/winntas/proddocs/inetconctservice/bcgstrtc.mspx (Internet Connection Services II).

Microsoft Corp., Internet Explorer 5 Corporate Deployment Guide—Appendix B:Enabling Connections with the Connection Manager Administration Kit, available at http://www.microsoft.com/technet/prodtechnol/ie/deploy/deploy5/appendb.mspx (IE5 Corporate Development).

Mark Minasi, *Mastering Windows NT Server 4* 1359–1442 (6th ed., Jan. 15, 1999)(Mastering Windows NT Server).

Hands On, Self–Paced Training for Supporting Verion 4.0 371–473 (Microsoft Press 1998) (Hands On).

Microsoft Corp., MS Point–to–Point Tunneling Protocol (Windows NT 4.0), available at http://www.microsoft.com/technet/archive/winntas/maintain/featusability/pptpwp3.mspx (MS PPTP).

Kenneth Gregg, et al., *Microsoft Windows NT Server Administrator's Bible* 173–206, 883–911, 974–1076 (IDG Books Worldwide 1999) (Gregg).

Microsoft Corp., Remote Access (Windows), available at http://msdn2.microsoft.com/en–us/library/bb545687 (VS.85,printer).aspx (Remote Access).

Microsoft Corp., Understanding PPTP (Windows NT 4.0), available at http://www.microsoft.com/technet/archive/winntas/plan/pptpudst.mspx (Understanding PPTP NT 4) (Although undated, this reference refers to the operation of prior art versions of Microsoft Windows such as Windows NT 4.0. Accordingly, upon information and belief, this reference is prior art to the patents–in–suit.).

Microsoft Corp., Windows NT 4.0: Virtual Private Networking, available at http://www.microsoft.com/technet/archive/winntas/deploy/confeat/vpntwk.mspx (NT4 VPN) (Although undated, this reference refers to the operation of prior art versions of Microsoft Windows such as Windows NT 4.0. Accordingly, upon information and belief, this reference is prior art to the patents–in–suit.).

Anthony Northrup, *NT Network Plumbing: Routers, Proxies, and Web Services* 299–399 (IDG Books Worldwide 1998) (Network Plumbing).

Microsoft Corp., Chapter 1—Introduction to Windows NT Routing with Routing and Remote Access Service, Available at http://www.microsoft.com/technet/archive/winntas/proddocs/ rras40/rrasch01.mspx (Intro to RRAS) (Although undated, this reference refers to the operation of prior art versions of Microsoft Windows such as Windows NT 4.0. Accordingly, upon information and belief, this reference is prior art to the patents–in–suit.) 13.

Microsoft Corp., Windows NT Server Product Documentation: Chapter 5—Planning for Large–Scale Configurations, available at http://www.microsoft.com/technet/archive/winntas/proddocs/rras40/rrasch05.mspx (Large–Scale Configurations) (Although undated, this reference refers to the operation of prior art versions of Microsoft Windows such as Windows NT 4.0. Accordingly, upon information and belief, this reference is prior art to the patents–in–suit.).

F–Secure, F–Secure Evaluation Kit (May 1999) (FSECURE 00000003) (Evaluation Kit 3).

F–Secure, F–Secure NameSurfer (May 1999) (FSECURE 00000003) (NameSurfer 3).

F–Secure, F–Secure VPN Administrator's Guide (May 1999) (from FSECURE 00000003) (F–Secure VPN 3).

F–Secure, F–Secure SSH User's & Administrator's Guide (May 1999) (from FSECURE 00000003) (SSH Guide 3).

F–Secure, *F–Secure SSH2.0 for Windows NT and 95* (May 1999) (from FSECURE 00000003) (SSH 2.0 Guide 3).

F–Secure, *F–Secure VPN+ Administrator's Guide* (May 1999) (from FSECURE 00000003) (VPN+ Guide 3).

F. Secure, *F–Secure VPN+ 4.1* (1999) (from FSECURE 00000006) (VPN+ 4.1 Guide 6).

F–Secure, *F–Secure SSH* (1996) (from FSECURE 00000006) (F–Secure SSH 6).

F–Secure, *F–Secure SSH 2.0 for Windows NT and 95* (1998) (from FSECURE 00000006) (F–Secure SSH 2.0 Guide 6).

F–Secure, *F–Secure Evaluation Kit* (Sep. 1998) (FSECURE 00000009) (Evaluation Kit 9).

F–Secure, *F–Secure SSH User's & Administrator's Guide* (Sep. 1998) (from FSECURE 00000009) (SSH Guide 9).

F–Secure, *F–Secure SSH 2.0 for Windows NT and 95* (Sep. 1998) (from FSECURE 00000009) (F–secure SSH 2.0 Guide 9).

F–Secure, *F–Secure VPN+* (Sep. 1998) (from FSECURE 00000009) (VPN+ Guide 9).

F–Secure, *F–Secure Management Tools Administrator's Guide* (1999) (from FSECURE 00000003) (F–secure Management Tools).

F–Secure, *F–Secure Desktop, User's Guide* (1997) (from FSECURE 00000009) (F–secure Desktop User's Guide).

SafeNet, Inc., *VPN Policy Manager* (Jan. 2000) (VPN Policy Manager).

F–Secure, *F–Secure VPN+ for Windows NT 4.0* (1998) (from FSECURE 00000009) (F–secure VPN+).

IRE, Inc., *SafeNet/Soft–PK Version 4* (Mar. 28, 2000) (Soft–PK Version 4).

IRE/SafeNet Inc., *VPN Technologies Overview* (Mar. 28, 2000) (Safenet VPN Overview).

IRE, Inc., *SafeNet/Security Center Technical Reference Addendum* (Jun. 22, 1999) (Safenet Addendum).

IRE, Inc., *System Description for VPN Policy Manager and SafeNet/SoftPK* (Mar. 30, 2000) (VPN Policy Manager System Description).

IRE, Inc., About SafeNet/VPN Policy Manager (1999) (About Safenet VPN Policy Manager).
IRE, Inc., *SafeNet/VPN Policy Manager Quick Start Guide Version 1* (1999) (SafeNet VPN Policy Manager).
Trusted Information Systems, Inc., *Gauntlet Internet Firewall, Firewall Product Functional Summary* (Jul. 22, 1996) (Gauntlet Functional Summary).
Trusted Information Systems, Inc., *Running the Gauntlet Internet Firewall, An Administrator's Guide to Gauntlet Version 3.0* (May 31, 1995) (Running the Gauntlet Internet Firewall).
Ted Harwood, *Windows NT Terminal Server and Citrix Metaframe* (New Riders 1999) (Windows NT Harwood) 79.
Todd W. Matehrs and Shawn P. Genoway, *Windows NT Thing Client Solutions: Implemetning Terminal Server and Citrix MetaFrame* (Macmillan Technical Publishing 1999) (Windows NT Mathers).
Bernard Aboba et al., *Securing L2TP using IPSEC* (Feb. 2, 1999).
156. *Finding Your Way Through the VPN Maze* (1999) ("PGP").
Linux FreeS/WAN Overview (1999) (Linux FreeS/WAN) Overview).
TimeStep, *The Business Case for Secure VPNs* (1998) ("TimeStep").
WatchGuard Technologies, Inc., *WatchGuard Firebox System Powerpoint* (2000).
WatchGuard Technologies, Inc., *MSS Firewall Specifications* (1999).
WatchGuard Technologies, Inc., *Request for Information, Security Services* (2000).
WatchGuard Technologies, Inc., *Protecting the Internet Distributed Enterprise, White Paper* (Feb. 2000).
WatchGuard Technologies, Inc., *WatchGuard LiveSecurity for MSS Powerpoint* (Feb. 14, 2000).
WatchGuard Technologies, Inc., *MSS Version 2.5, Add–On for WatchGuard SOHO Releaset Notes* (Jul. 21, 2000).
Air Force Research Laboratory, *Statement of Work for Information Assurance System Architecture and Integration, PR No. N–8–6106* (Contract No. F30602–98–C–0012) (Jan. 29, 1998).
GTE Internetworking & BBN Technologies DARPA *Information Assurance Program Integrated Feasibility Demonstration* (*IFD*) 1.2 Report, Rev. 1.0 (Sep. 21, 1998).
BBN Information Assurance Contract, *TIS Labs Monthly Status Report* (Mar. 16–Apr. 30, 1998).
DARPA, *Dynamic Virtual Private Network* (*VPN*) *Powerpoint*.
GTE Internetworking, *Contractor's Program Progress Report* (Mar. 16–Apr. 30, 1998).
Darrell Kindred, *Dynamic Virtual Private Networks* (*DVPN*) *Countermeasure Characterization* (Jan. 30, 2001).
Virtual Private Networking Countermeasure Characterization (Mar. 30, 2000).
Virtual Private Network Demonstration (Mar. 21, 1998).
Information Assurance/NAI Labs, *Dynamic Virtual Private Networks* (*VPNs*) *and Integrated Security Management* (2000).
Information Assurance/NAI Labs, *Create/Add DVPN Enclave*(2000).
NAI Labs, *IFE 3.1 Integration Demo* (2000).
Information Assurance, *Science Fair Agenda* (2000).
Darrell Kindred et al., *Proposed Threads for IFE 3.1* (Jan. 13, 2000).
IFE 3.1 Technology Dependencies (2000).
IFE 3.1 Topology (Feb. 9, 2000).
Information Assurance, *Informatiion Assurance Integration: IFE 3.1, Hypothesis & Thread Development* (Jan. 10–11, 2000).
Information Assurance/NAI Labs, *Dynamic Virtual Private Networks Presentation* (2000).
Information Assurance/NAI Labs, *Dynamic Virtual Private Networks Presentation v.2* (2000).
Information Assurance/NAI Labs, *Dynamic Virtual Private Networks Presentation v.3* (2000).
T. Braun et al., *Virtual Private Network Architecture,* Charging and Accounting Technology for the Internet (Aug. 1, 1999) (VPNA).
Network Associates Products—*PGP Total Network Security Suite, Dynamic Virtual Private Networks* (1999).
Microsoft Corporation, Microsoft Proxy Server 2.0 (1997) (Proxy Server 2.0, Microsoft Prior Art VPN Technology).
David Johnson et al., *A Guide To Microsoft Proxy Server 2.0* (1999) (Johnson, Microsoft Prior Art VPN Technology).
Microsoft Corporation, *Setting Server Parameters* (1997 (Proxy Server 2.0 CD labeled MSFTVX00157288) (Setting Server Parameters, Microsoft Prior Art VPN Technology).
Kevin Schuler, *Microsoft Proxy Server 2* (1998) (Schuler, Microsoft Prior Art VPN Technology).
Erik Rozell et al., *MCSE Proxy Server 2 Study Guide* (1998) (Rozell, Microsoft Prior 15 Art VPN Technology.
M. Shane Stigler & Mark A. Linsenbardt, *IIS 4 and Proxy Server 2* (1999) (Stigler, Microsoft Prior Art VPN Technology).
David G. Schaer, *MCSE Test Success: Proxy Server 2* (1998) (Schaer, Microsoft Prior Art VPN Technology).
John Savill, *The Windows NT and Windows 2000 Answer Book* (1999) (Savill, Microsoft Prior Art VPN Technology).
Network Associates *Gauntlet Firewall Global Virtual Private Network User's Guide for Windows NT Version 5.0* (1999) (Gauntlet NT GVPN, GVPN).
Network Associates *Gauntlet Firewall For UNIX Global Virtual Private Network User's Guide Version 5.0* (1999) (Gauntlet Unix GVPN, GVPN).
File History for U.S. Appl. No. 09/653,201, Applicant(s): Whittle Bryan, et al., filed Aug. 31, 2000.
AutoSOCKS v2.1, Datasheet, http://web.archive.org/web/19970212013409/www.aventail.com/prod/autoskds.html.
Ran Atkinson, *Use of DNS to Distribute Keys,* Sep. 7, 1993, http://ops.ietf.org/lists/namedroppers/namedroppers.199x/msg00945.html.
FirstVPN Enterprise Networks, Overview.
Chapter 1: Introduction to Firewall Technology, Administration Guide: Dec. 19, 2007, http://www.books24x7.com/book/id_762/viewer_r.asp?bookid=762&chunked=41065062.
The TLS Protocol Version 1.0; Jan. 1999; p. 65 of 71.
Elizabeth D. Zwicky, et al., Building Internet Firewalls, 2nd Ed.
Virtual Private Networks—Assured Digital Incorporated—ADI 4500; http://web.archive.org/web/1990224050035/www.assured–digital.com/products/prodvpn/adia4500.htm.
Accessware—The Third Wave in Network Security, Conclave from Internet Dynamics; http://web.archive.org/web/11980210013830/interdyn.com/Accessware.html.
Extended System Press Release, Sep. 2, 1997; *Extended VPN Uses The Internet to Create Virtual Private Networks,* www.extendedsystems.com.

Socks Version 5; Executive Summary; http://web.archive.org/web/199970620031945/www.aventail.com/educate/whitepaper/sockswp.html.
Internet Dynamics First to Ship Integrated Security Solutions for Enterprise Intranets and Extranets; Sep. 15, 1997; http://web.archive.org/web/19980210014150/interdyn.com.
E–mails from various individuals to Linux IPsec re:DNS–LDAP Splicing.
Microsoft Corporation's Fifth Amended Invalidity Contentions dated Sep. 18, 2009, *VirnetX Inc. and Science Applications International Corp.* v. *Microsoft Corporation* and invalidity claim charts for U.S. Patent Nos. 7,188,180 and 6,839,759.
The IPSEC Protocol as described in Atkinson, et al., "Security Architecture for the Internet Protocol," Networking Working Group, RFC 2401 (Nov. 1998) ("RFC 2401"); http://web.archive.org/web/19991007070353/http://www.imib.med.tu–dresden.de/imib/Internet/Literatur/ipsec–docu_eng.html.
S. Kent and R. Atkinson, "IP Authentication Header," RFC 2402 (Nov. 1998); http://web.archive.org/web/19991007070353/http://www.imib.med.tu–dresden.de/imib/Internet/Literatur/ipsec–docu_eng.html.
C. Madson and R. Glenn, "The Use of HMAC–MD5–96 within ESP and AH," RFC 2403 (Nov. 1998); http://web.archive.org/web/19991007070353/http://www.imib.med.tu–dresden.de/imib/Internet/Literatur/ipsec–docu_eng.html.
C. Madson and R. Glenn, "The Use HMAC–SHA–1–96 with ESP and AH," RFC 2404 (Nov. 1998); http://web.archive.org/web/19991007070353/http://www.imib.med.tu–dresden.de/imib/Internet/Literatur/ipsec–docu_eng.html.
C. Madson and N. Doraswamy, "The ESP DES–CBC Cipher Algorithm With Explicit IV", RFC 2405 (Nov. 1998); http://web.archive.org/web/19991007070353/http://www.imib.med.tu–dresden.de/imib/Internet/Literatur/ipsec–docu_eng.html.
S.Kent and R. Atkinson, "IP Encapsulating Security Payload (ESP)," RFC 2406 (Nov. 1998); http://web.archive.org/web/19991007070353/http://www.imib.med.tu–dresden.de/imib/Internet/Literatur/ipsec–docu_eng.html.
Derrell Piper, "The Internet IP Security Domain of Interpretation for ISAKMP," RFC 2407 (Nov. 1998); http://web.archive.org/web/19991007070353/http://www.imib.med.tu–dresden.de/imib/Internet/Literatur/ipsec–docu_eng.html.
Douglas Maughan, et al, "Internet Security Association and Key Management Protocol (ISAKMP)," RFC 2408 (Nov. 1998); http://web.archive.org/web/19991007070353/http://www.imib.med.tu–dresden.de/imib/Internet/Literatur/ipsec–docu_eng.html.
D. Harkins and D. Carrell, "The Internet Key Exchange (IKE)," RFC 2409 (Nov. 1998); http://web.archive.org/web/19991007070353/http://www.imib.med.tu–dresden.de/imib/Internet/Literatur/ipsec–docu_eng.html.
R. Glenn and S. Kent, "The NULL Encryption Algorithm and Its Use With IPsec," RFC 2410 (Nov. 1998); http://web.archive.org/web/19991007070353/http://www.imib.med.tu–dresden.de/imib/Internet/Literatur/ipsec–docu_eng.html.
R. Thayer, et al., "IP Security Document Roadmap," RFC 2411 (Nov. 1998); http://web.archive.org/web/19991007070353/http://www.imib.med.tu–dresden.de/imib/Internet/Literatur/ipsec–docu_eng.html.
Hilarie K. Orman, "The Oakley Key Determination Protocol," RFC 2412 (Nov. 1998) in combination with J.M. Galvin, "Public Key Distribution with Secure DNS," Proceedings of the Sixth USENIX UNIX Security Symposium, San Jose California (Jul. 1996) ("Galvin").
David Kosiur, "Building and Managing Virtual Private Networks" (1998).
P. Mockapetris, "Domain Names—Implementation and Specification," Network Working Group, RFC 1035 (Nov. 1987).
Request for Inter Partes Reexamination of Patent No. 7,188,180, dated Nov. 25, 2009.
Exhibit 2 "Aventail Connect v3.1/v2.6 Administrator's Guide", 120 pages, 1996–1999.
Exhibit 3A, "Gauntlet Firewall for Windows", pp. 1–137, 1998–1999.
Exhibit 3B, "Gauntlet Firewall for Windows", pp. 138–275, 1998–1999.
Exhibit 4, "Kosiur", Building and Managing VPNs, pp. 1–396, 1998.
Exhibit 5, Building a Microsoft VPN; A comprehensive Collection of Microfoft Resources, pp. 1–216.
Exhibit 6, Windows NT Server, Virtual Private Network; An Overview, pp. 1–26, 1998.
Exhibit 7, "Networking Working Group Request for Comments: 1035" pp. 1–56, 1987.

INTER PARTES REEXAMINATION CERTIFICATE ISSUED UNDER 35 U.S.C. 316

THE PATENT IS HEREBY AMENDED AS INDICATED BELOW.

Matter enclosed in heavy brackets [ ] appeared in the patent, but has been deleted and is no longer a part of the patent; matter printed in italics indicates additions made to the patent.

AS A RESULT OF REEXAMINATION, IT HAS BEEN DETERMINED THAT:

The patentability of claims **1-10** and **12** is confirmed.

New claim **18** is added and determined to be patentable.

Claims **11** and **13-17** were not reexamined.

18. A method of transparently creating a virtual private network (VPN) between a client computer and a target computer, comprising the steps of:

(1) generating from the client computer a Domain Name Service (DNS) request that requests an IP address corresponding to a domain name associated with the target computer;

(2) determining whether the DNS request transmitted in step (1) is requesting access to a secure web site; and

(3) in response to determining that the DNS request in step (2) is requesting access to a secure target web site, automatically initiating the VPN between the client computer and the target computer, wherein:

steps (2) and (3) are performed at a DNS server separate from the client computer, and step (3) comprises the step of, prior to automatically initiating the VPN between the client computer and the target computer, determining whether the client computer is authorized to resolve addresses of non secure target computers and, if not so authorized, returning an error from the DNS request.

* * * * *

CERTIFICATE OF SERVICE

I, Naveen Modi, hereby certify that on August 22, 2022, the foregoing brief was filed using the Court's CM/ECF system and a copy served on the parties' counsel of record via ECF.

Date: August 22, 2022 BY: /s/Naveen Modi

Naveen Modi
PAUL HASTINGS LLP
2050 M Street, N.W.
Washington, D.C. 20036
Tel.: (202) 551-1700
Fax: (202) 551-1705

CERTIFICATE OF COMPLIANCE WITH TYPE-VOLUME LIMITATION, TYPEFACE REQUIREMENTS AND TYPE STYLE REQUIREMENTS

1. This brief complies with the type-volume limitation of Federal Circuit Rule 32 and Federal Circuit Rule 28.1(b)(1)(A).

The brief contains 9,657 words, excluding the parts of the brief exempted by Federal Rule of Appellate Procedure 32(f) and Federal Circuit Rule 32(b).

2. This brief complies with the typeface requirements of Federal Rule of Appellate Procedure 32(a)(5) and the type style requirements of Federal Rule of Appellate Procedure 32(a)(6).

The brief has been prepared in a proportionally spaced typeface using MS Word 2013 in a 14 point Times New Roman font.

Date: August 22, 2022 BY: /s/Naveen Modi

Naveen Modi
PAUL HASTINGS LLP
2050 M Street, N.W.
Washington, D.C. 20036
Tel.: (202) 551-1700
Fax: (202) 551-1705